Relational Database Design using E/R

Introduction to Database Management

CS348 Fall 2022

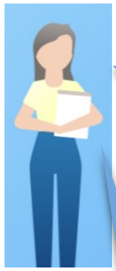
Announcements (Tue. Oct 4)

- Assignments
  - Release assignment 1 grade (week 6)
  - Release assignment 2 (lectures 5-10) by tonight

- Project
  - Ensure correct enrollment of teams on Learn
  - Receive a note from TA on Learn by Fri, including
    - Name, email, their github id (if needed)
  - Start working on your project
  - Set mini-milestones

- Start earlier ☺

Motivating Example



I want to have a registrar's database. Can you help?

It has these requirements …

Zero or more sections of a course are offered each term. Courses have names and numbers. In each term, the sections of each course are numbered starting with 1.
Most course sections are taught on-site, but a few are taught at off-site locations.
Students have student numbers and names. Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.
Up to 50 students may be registered for a course section. Sections with 5 or fewer students are cancelled.
A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.

I know how to use SQL now!

What tables do you want me to create? What are the primary keys, constraints, queries, … ..?

We still need to learn about database design ☺

Database Design

Step 1: Understand the real-world domain being modeled

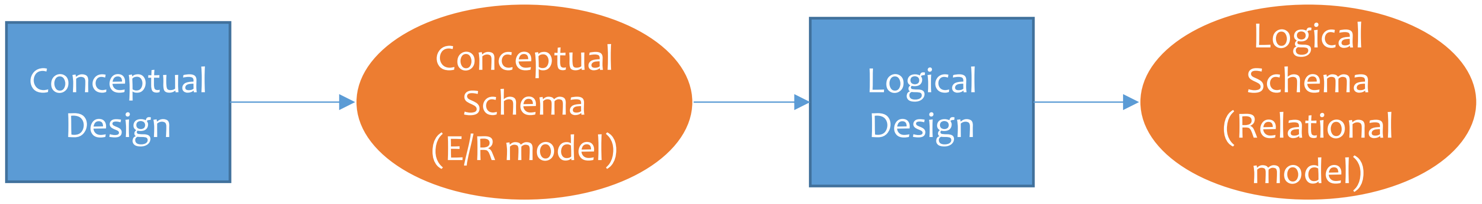
→Specify it using a database design model

- E.g., Entity/Relationship (E/R) model, Object Definition Language (ODL), UML (Unified Modeling Language)

Step 2: Translate specification to the data model of DBMS

- Relational, XML, object-oriented, etc.

→ Create DBMS schema


Conceptual Design
Conceptual Schema (E/R model)
Logical Design
Logical Schema (Relational model)

Database Design

- Entity-Relationship (E/R) model
- Translating E/R to relational schema
- Relational design principles

this week

But what about ORM?

- Automatic object-relational mappers are made popular by rapid Web development frameworks

- But you still need designer discretion in all but simple cases

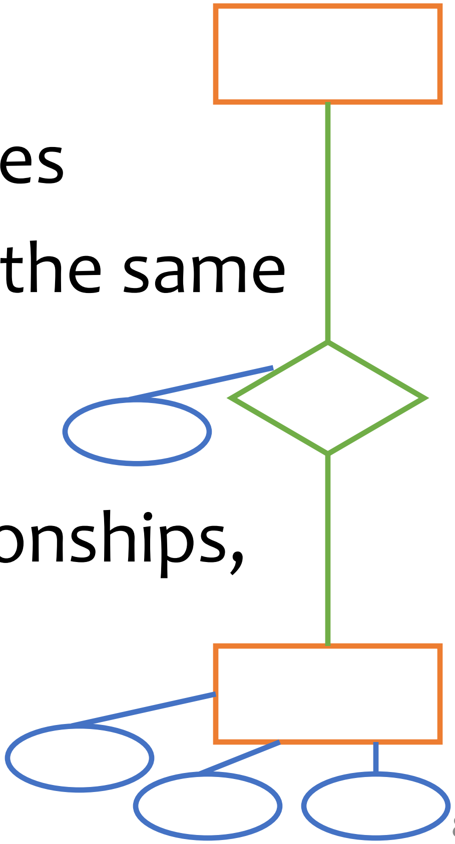
- Each language/library has its own syntax for creating schema and for querying/modifying data
  - Quirks and limitations cause portability problems
  - They are not necessarily easier to learn than SQL

Entity-relationship (E/R) model

- Historically and still very popular
- Primarily a design model—not directly implemented by DBMS
- Designs represented by E/R diagrams
  - We use the style of E/R diagram covered by the textbook book; there are other styles/extensions
  - Very similar to UML diagrams

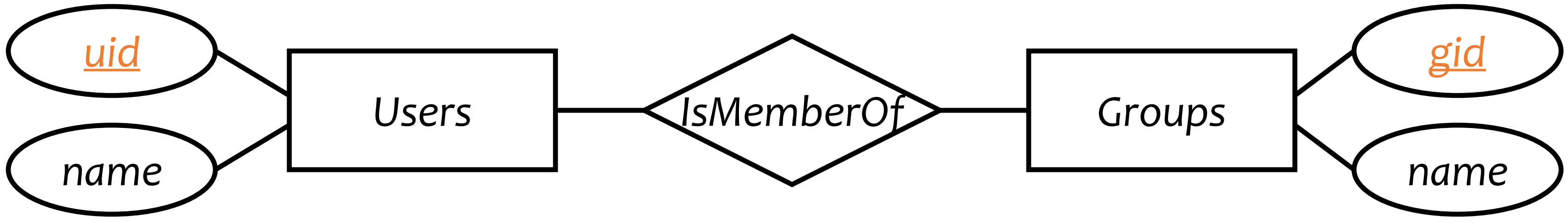
E/R basics

- Entity: a "thing," like an object
- Entity set: a collection of things of the same type, like a relation of tuples or a class of objects
    - Represented as a rectangle
- Relationship: an association among entities
- Relationship set: a set of relationships of the same type (among same entity sets)
    - Represented as a diamond
- Attributes: properties of entities or relationships, like attributes of tuples or objects
    - Represented as ovals



An example E/R diagram

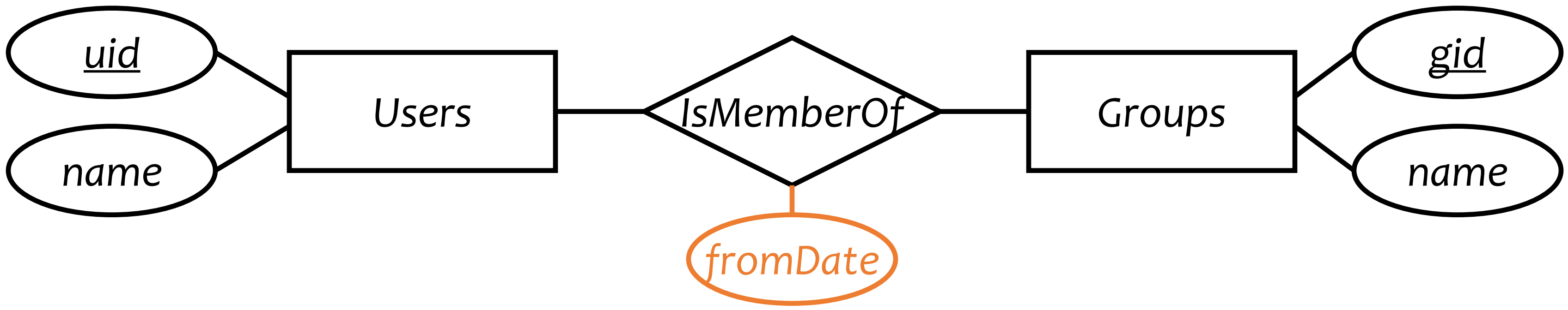
- Users are members of groups



- A key of an entity set is represented by underlining all attributes in the key
  - A key is a set of attributes whose values can belong to at most one entity in an entity set—like a key of a relation

Attributes of relationships

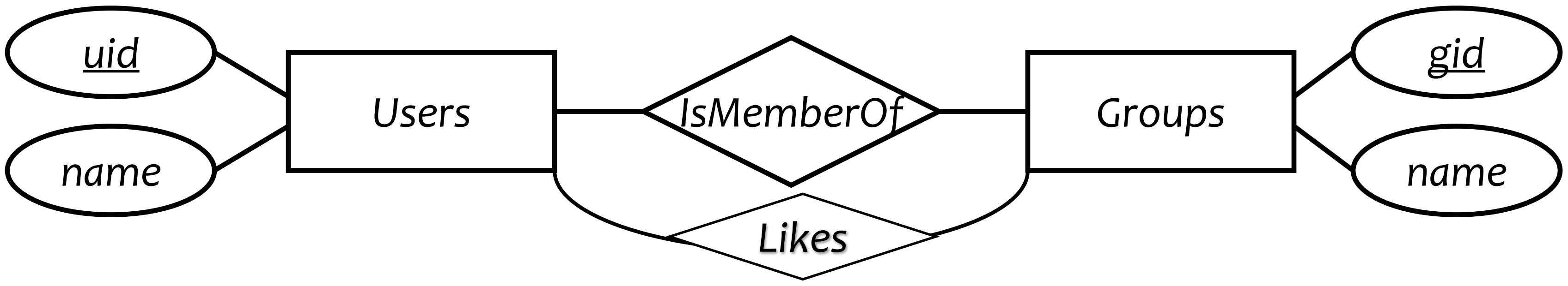
- Example: a user belongs to a group since a particular date



- Where do the dates go?
  - With *Users*?
    - But a user can join multiple groups on different dates
  - With *Groups*?
    - But different users can join the same group on different dates
  - With *IsMemberOf*!

More on relationships

- There could be multiple relationship sets between the same entity sets
  - Example: *Users IsMemberOf Groups*; *Users Likes Groups*
- In a relationship set, each relationship is uniquely identified by the entities it connects
  - Example: Between Bart and "Dead Putting Society", there can be at most one *IsMemberOf* relationship and at most one *Likes* relationship



More on relationships

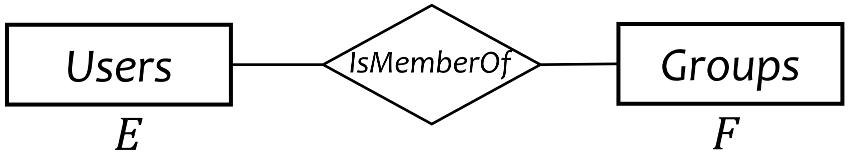
- There could be multiple relationship sets between the same entity sets
  - Example: *Users IsMemberOf Groups*; *Users Likes Groups*
- In a relationship set, each relationship is uniquely identified by the entities it connects
  - Example: Between Bart and "Dead Putting Society", there can be at most one *IsMemberOf* relationship and at most one *Likes* relationship

  ☞What if Bart joins DPS, leaves, and rejoins? How can we modify the design to capture historical membership information?

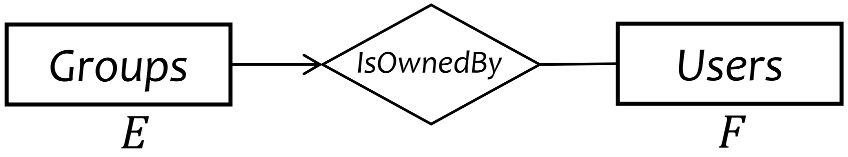
    ☞Make an entity set of *MembershipRecords*

Multiplicity of relationships

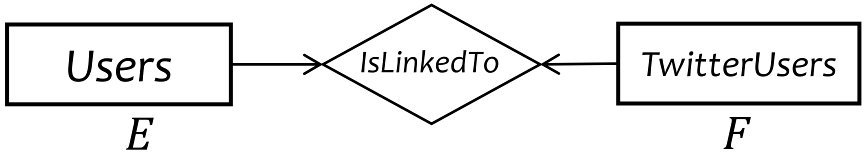
- E and F: entity sets
- Many-many: Each entity in E is related to 0 or more entities in F and vice versa
  - Example:



- Many-one: Each entity in E is related to 0 or 1 entity in F, but each entity in F is related to 0 or more in E
  - Example:



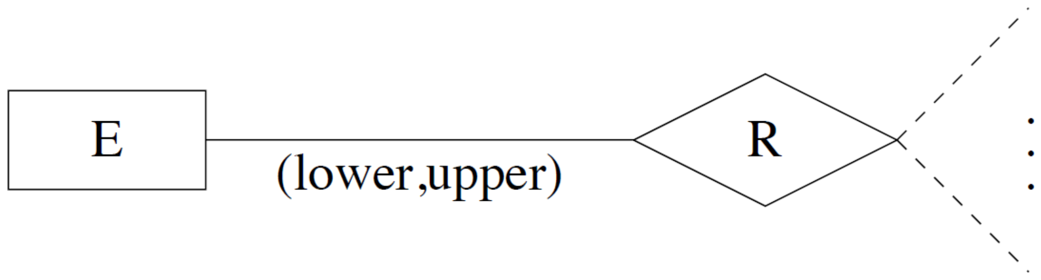
- One-one: Each entity in E is related to 0 or 1 entity in F and vice versa
  - Example:



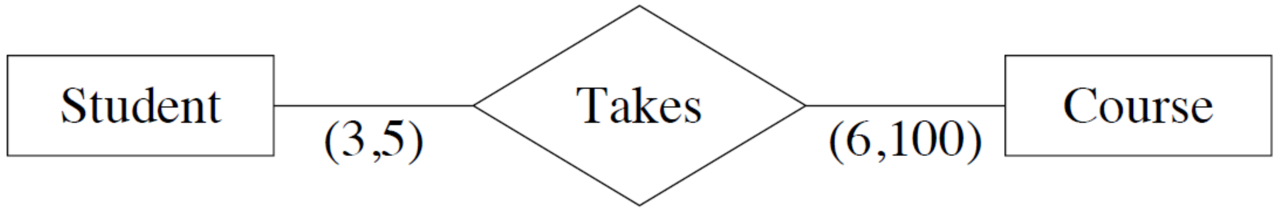
- "One" (0 or 1) is represented by an arrow $\longrightarrow$

General cardinality constraints

- General cardinality constraints determine lower and upper bounds on the number of relationships of a given relationship set in which a component entity may participate



- Example:

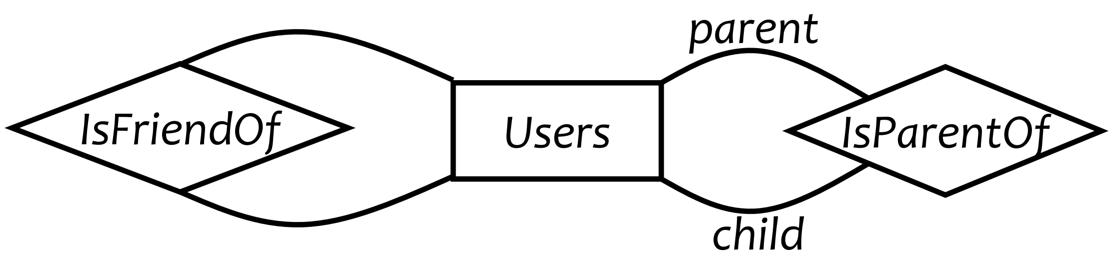


Roles in relationships

- An entity set may participate more than once in a relationship set

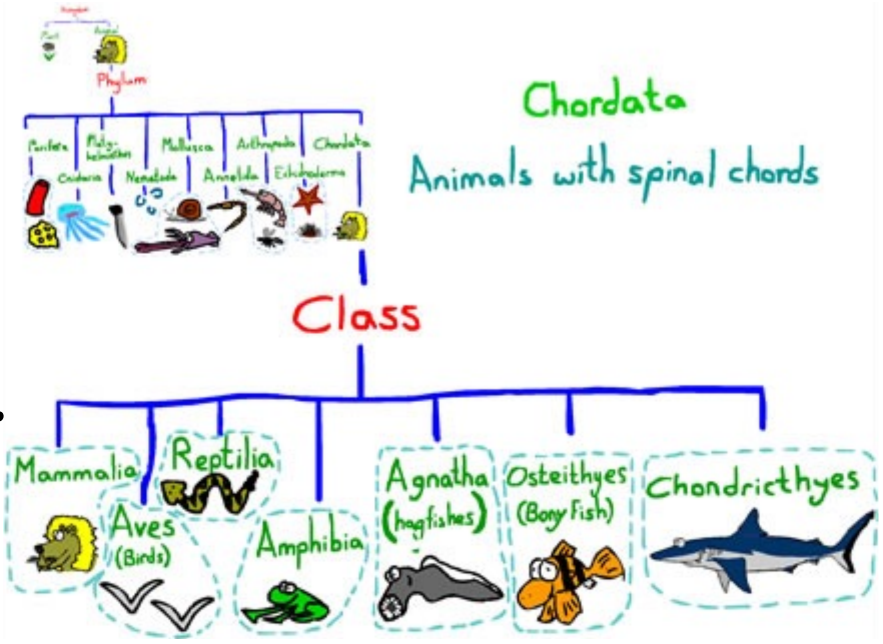
☞ May need to label edges to distinguish roles

- Examples
  - Users may be parents of others; label needed
  - Users may be friends of each other; label not needed



Next: two special relationships

… is part of/belongs to …



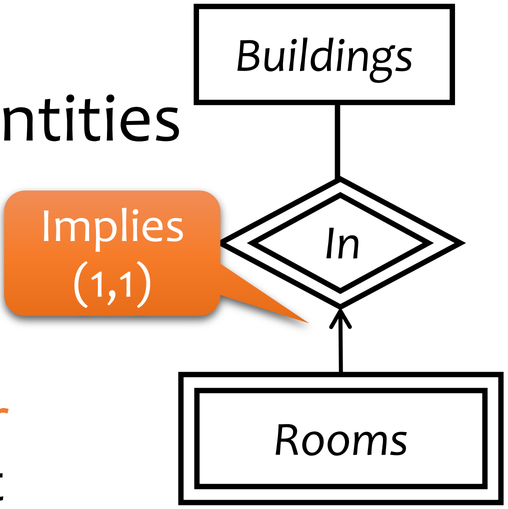
… is a kind of …

http://blogs.library.duke.edu/renovation/files/2012/08/Rubenstein-Library-First-Floor-Floorplan.jpg
http://www.sharky-jones.com/Sharkyjones/Artwork/taxonomy%20artwork/Class1.jpg

Weak entity sets

- If entity E is existence dependent on entity F, then
  - F is a dominant entity
  - E is a subordinate entity
  - Example: *Rooms* inside *Buildings* are partly identified by *Buildings*' name

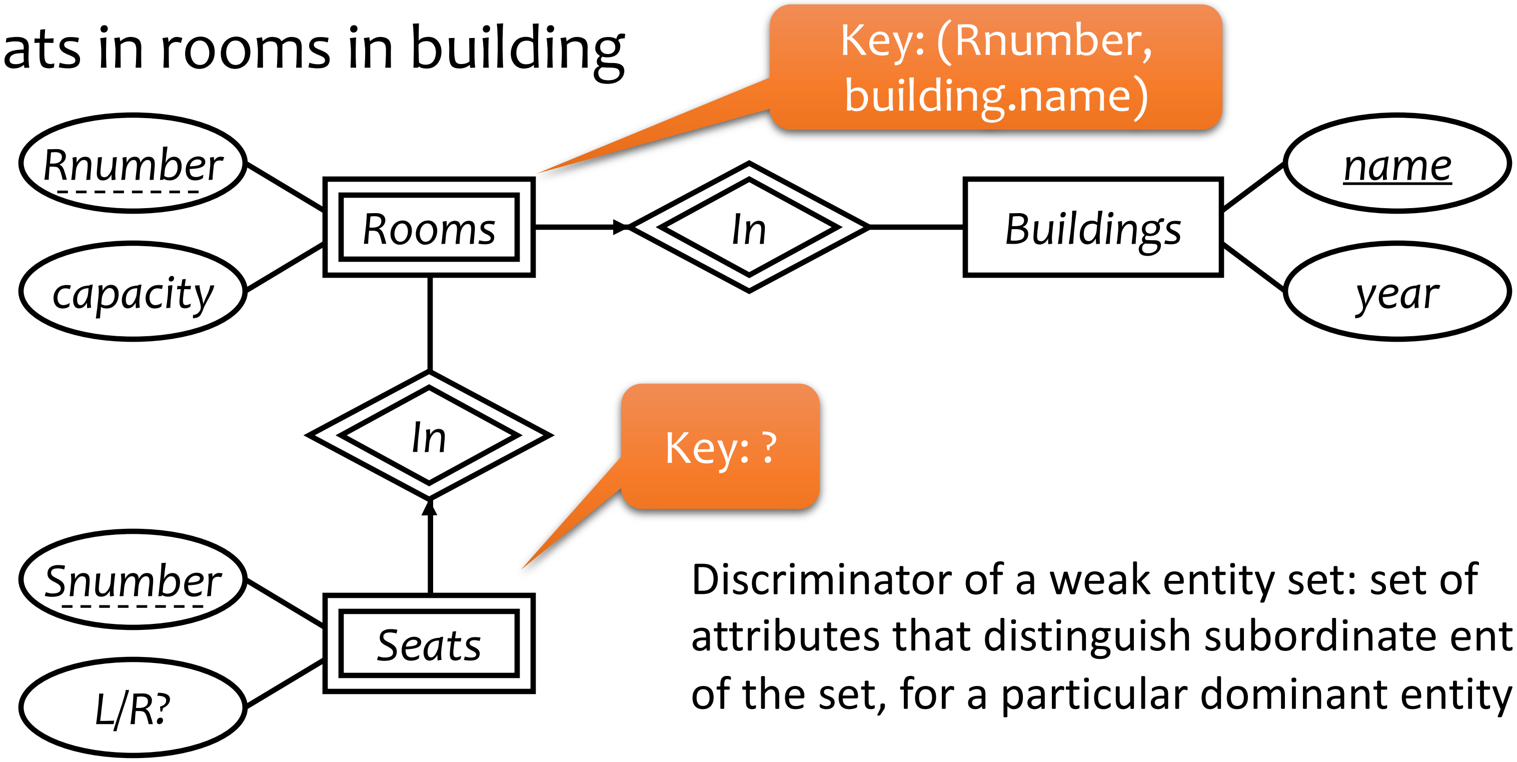
- Weak entity set: containing subordinate entities
  - Drawn as a double rectangle
  - The relationship sets are called supporting relationship sets, drawn as double diamonds
  - A weak entity set must have a many-to-one or one-to-one relationship to a distinct entity set



- Strong entity set: containing no subordinate entities

Weak entity set examples

- Seats in rooms in building

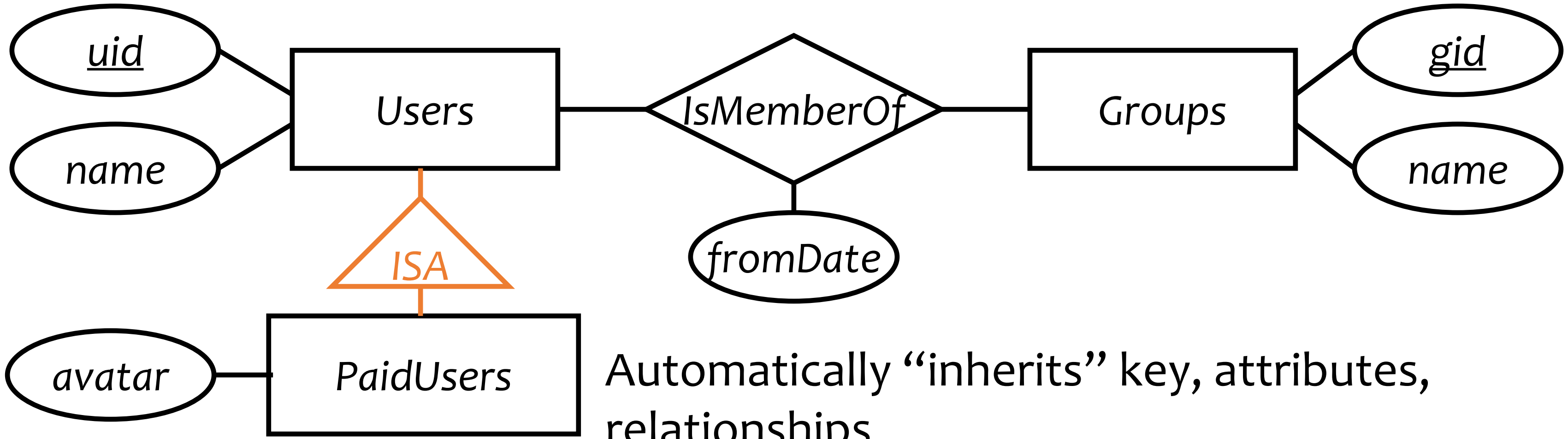


Discriminator of a weak entity set: set of attributes that distinguish subordinate entities of the set, for a particular dominant entity

- Attributes of weak entity sets only form key relative to a given dominant entity → discriminator (dotted underline)
- Primary key of a weak entity set: discriminator + primary key of entity set for dominant entities

ISA relationships

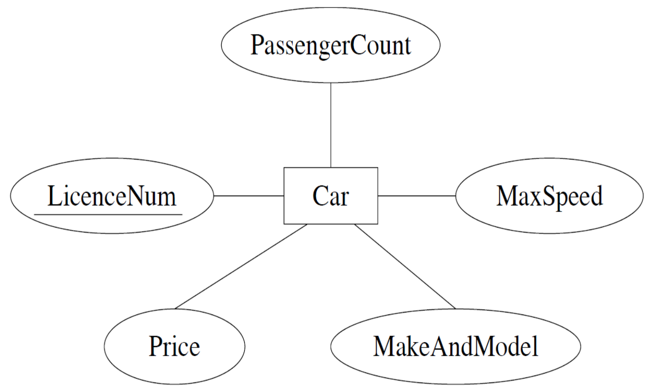
- Similar to the idea of subclasses in object-oriented programming: subclass = special case, fewer entities, and possibly more properties
  - Represented as a triangle (direction is important)
- Example: paid users are users, but they also get avatars (yay!)

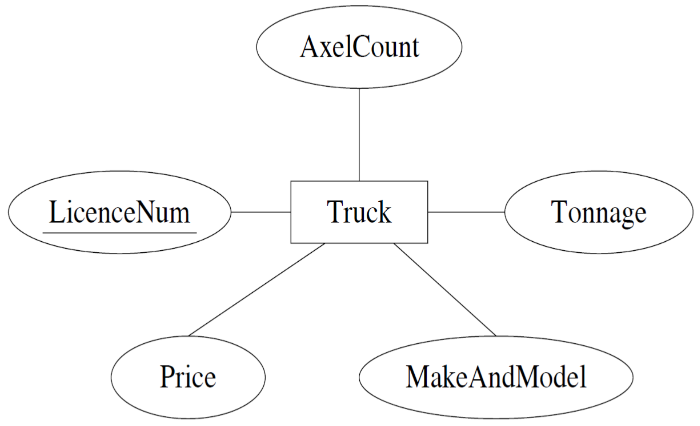


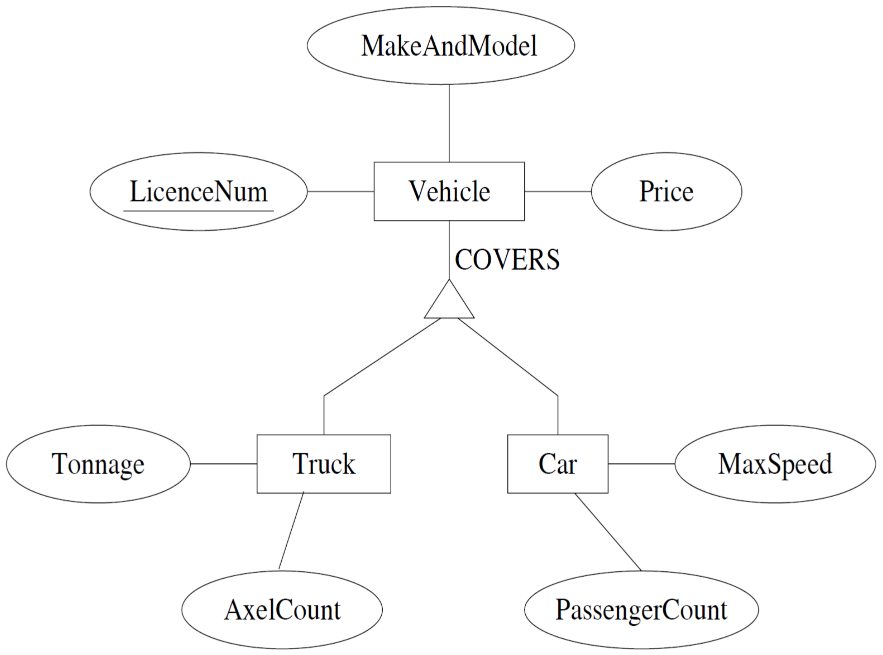
Automatically "inherits" key, attributes, relationships

Other extensions to E/R models

- Generalization: several entity sets can be abstracted by a more general entity set
  - Example: “ a vehicle abstracts the notion of a car and a truck”

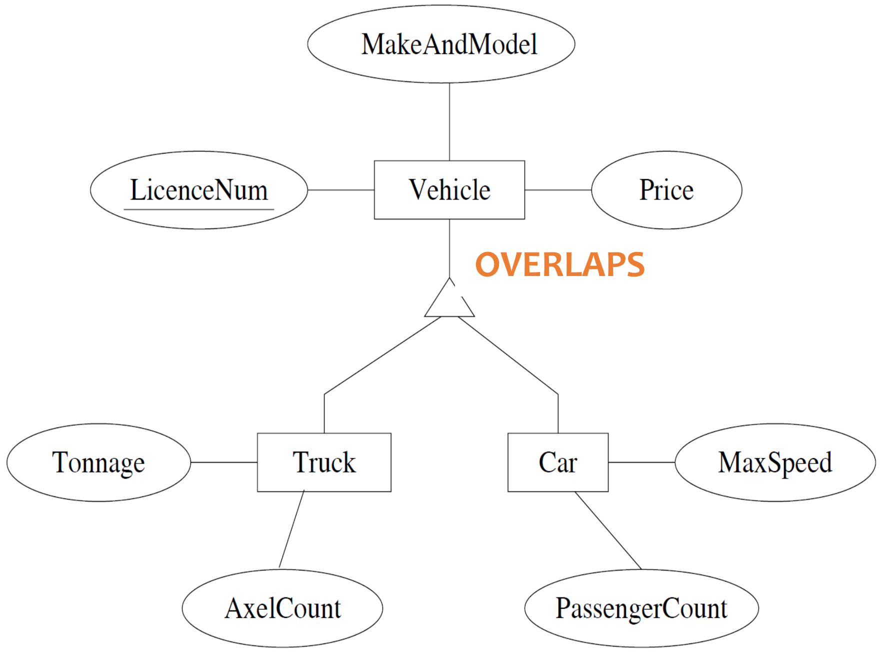






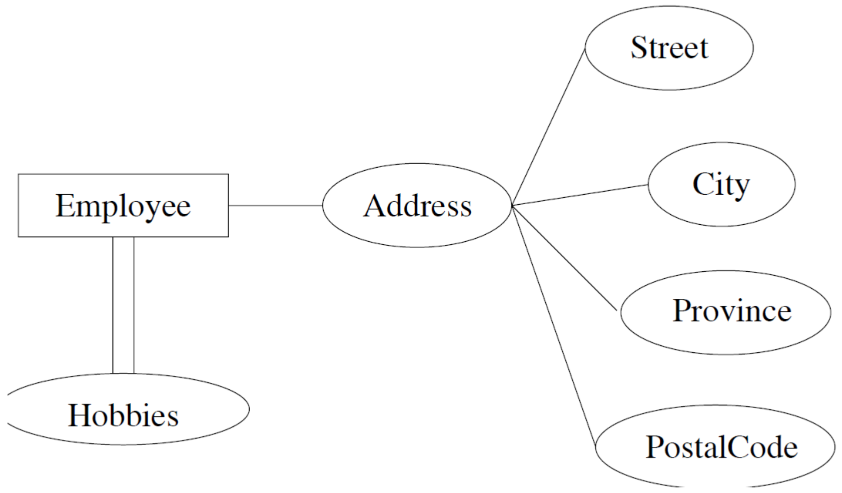
Other extensions to E/R models

- Specialized entity sets are usually disjoint but can be declared to have entities in common
- By default, specialized entity sets are disjoint.
  - Example: We may decide that nothing is both a car and a truck.
  - However, we can declare them to overlap (to accommodate utility vehicles, perhaps).



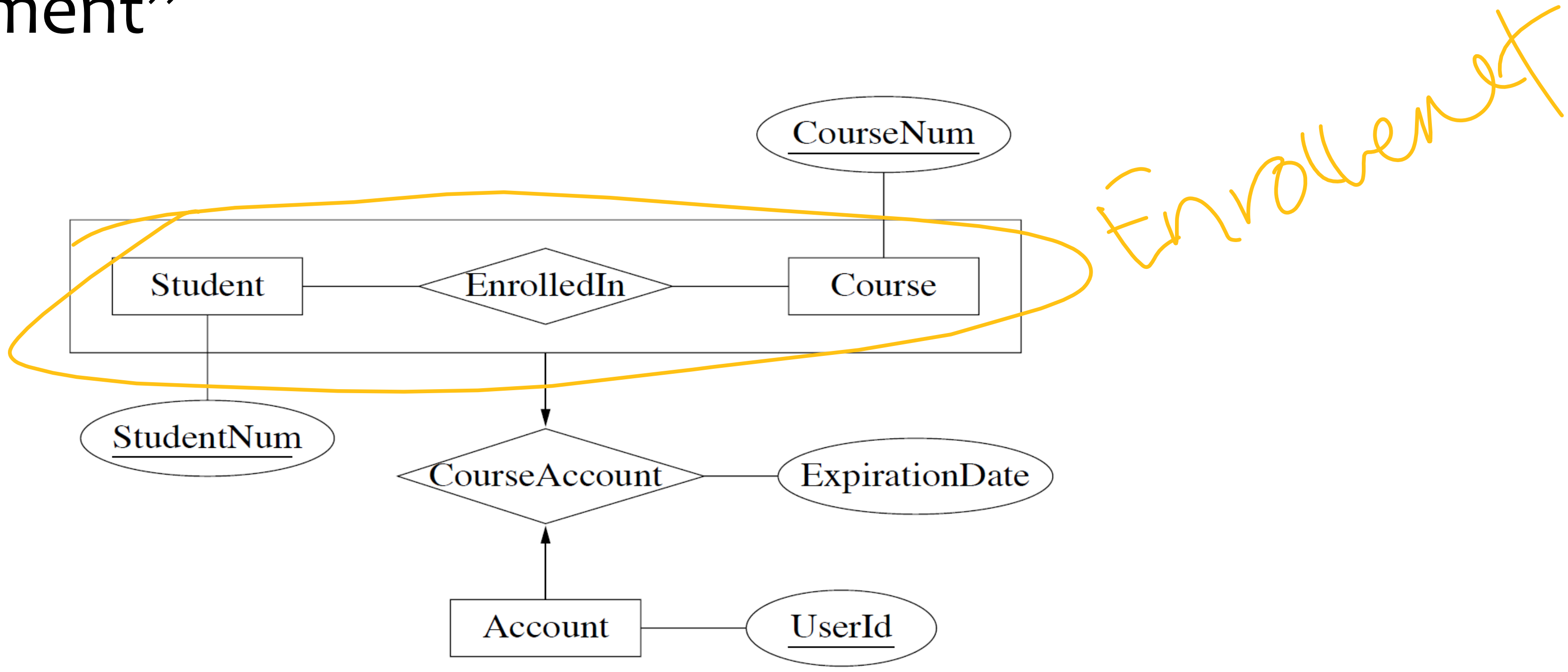
Other extensions to E/R models

- Structured attributes:
  - Composite attributes: composed of fixed number of other attributes
    - E.g. Address
  - Multi-valued attributes: attributes that are set-valued
    - e.g. Hobbies (double edges)



Other extensions to E/R models

- Aggregation: relationships can be viewed as high-level entities
- Example: "accounts are assigned to a given student enrollment"



Summary of E/R concepts

- Entity sets
  - Keys
  - Weak entity sets
- Relationship sets
  - Attributes of relationships
  - Multiplicity
  - Roles
  - Supporting relationships (related to weak entity)
  - ISA relationships
- Other extensions:
  - Generalization
  - Structured attributes
  - Aggregation

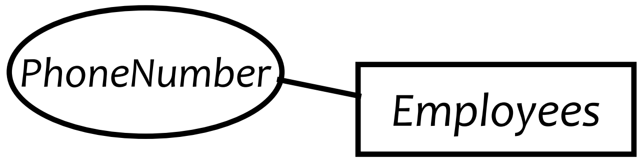
Designing an E/R schema

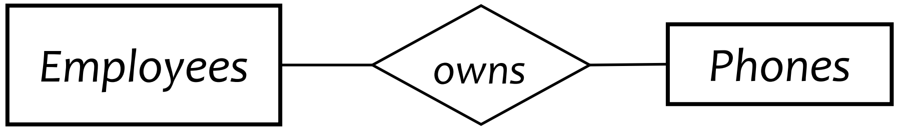
- Usually many ways to design an E-R schema

- Points to consider
  - use attribute or entity set?
  - use entity set or relationship set?
  - degrees of relationships?
  - extended features?

Attributes or Entity Sets?

- Example: How to model employees' phones?



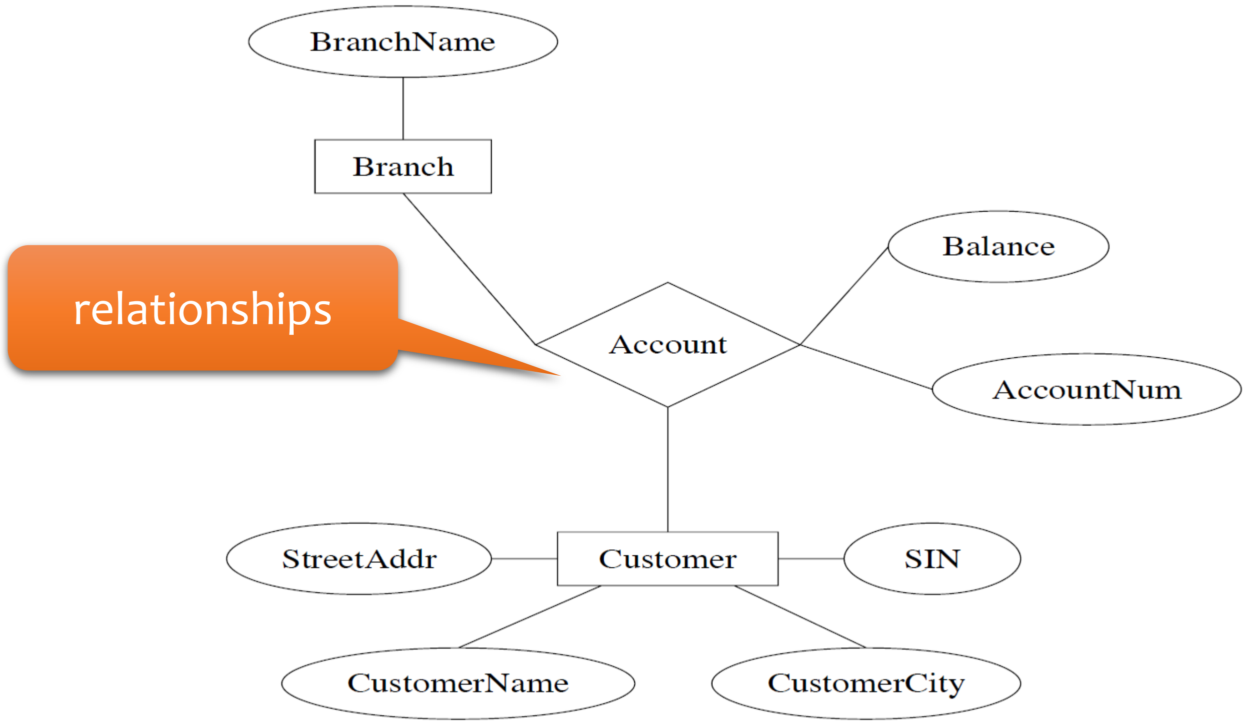


- Rules of thumb:
  - Is it a separate object?
  - Do we maintain information about it?
  - Can several of its kind belong to a single entity?
  - Does it make sense to delete such an object?
  - Can it be missing from some of the entity set's entities?
  - Can it be shared by different entities?

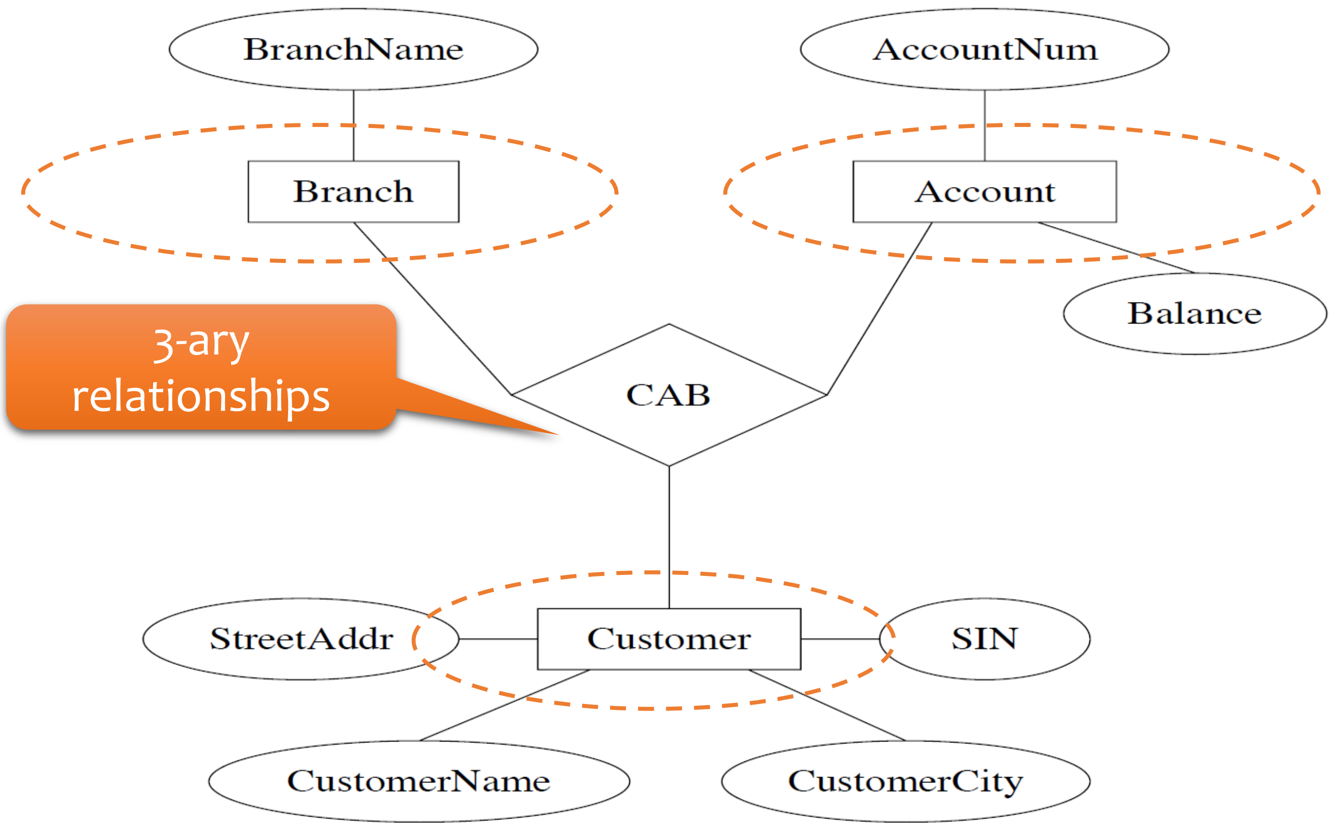
→ An affirmative answer to any of the above suggests a new entity set.

Entity Sets or Relationships?

- Instead of representing accounts as entities, we could represent them as relationships

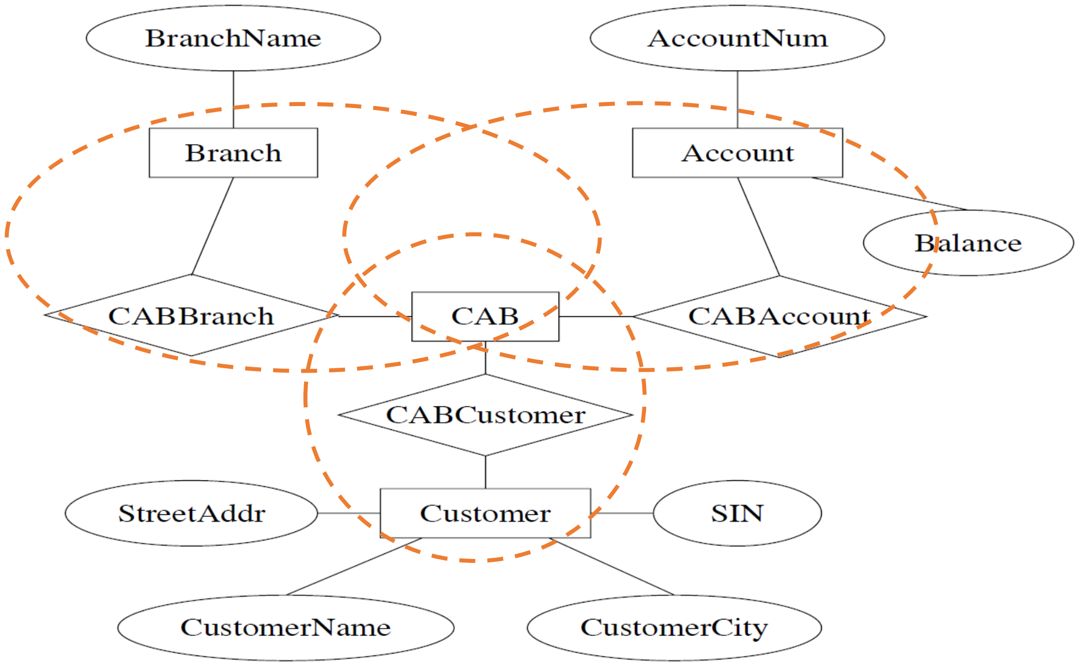


Binary vs. N-ary Relationships?



Binary vs. N-ary Relationships (cont'd)

- We can always represent a relationship on n entity sets with n binary relationships

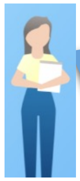


A simple methodology

1. Recognize entity sets
2. Recognize relationship sets and participating entity sets
3. Recognize attributes of entity and relationship sets
4. Define relationship types and existence dependencies
5. Define general cardinality constraints, keys and discriminators
6. Draw diagram

- For each step, maintain a log of assumptions motivating the choices, and of restrictions imposed by the choices

Case study 1



Design a database representing cities, counties, and states

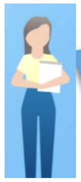
- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

Assume the following:

- Names of states are unique
- Names of counties are only unique within a state
- Names of cities are only unique within a county
- A city is always located in a single county
- A county is always located in a single state

What are the entity sets, relationship sets, and their attributes? What are the types of relationships and cardinality constraints, keys, discriminators?

Case study 1

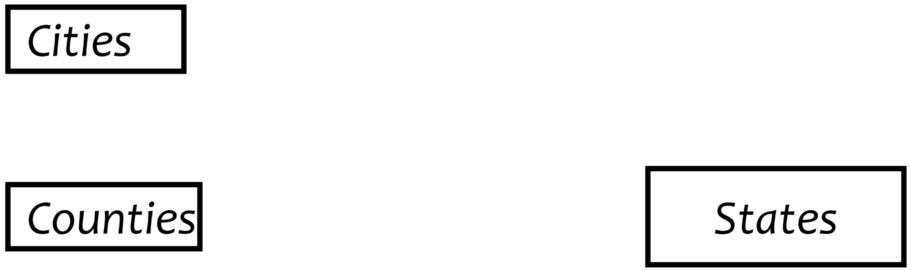


Design a database representing cities, counties, and states

- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

Assume the following:

- Names of states are unique
- Names of counties are only unique within a state
- Names of cities are only unique within a county
- A city is always located in a single county
- A county is always located in a single state



What are my entity sets? (slide 26)

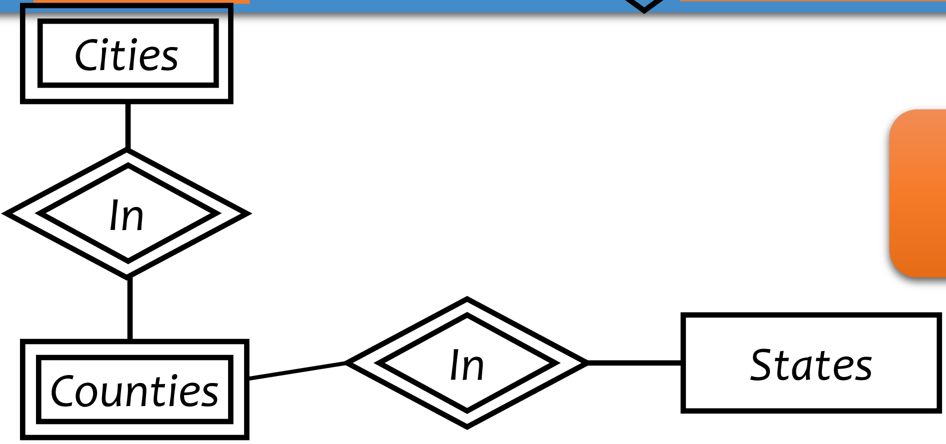
Case study 1

Design a database representing cities, counties, and states

- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

Assume the following:

- Names of states are unique
- Names of counties are only unique within a state
- Names of cities are only unique within a county
- A city is always located in a single county
- A county is always located in a single state



What are my relationship sets?

Case study 1

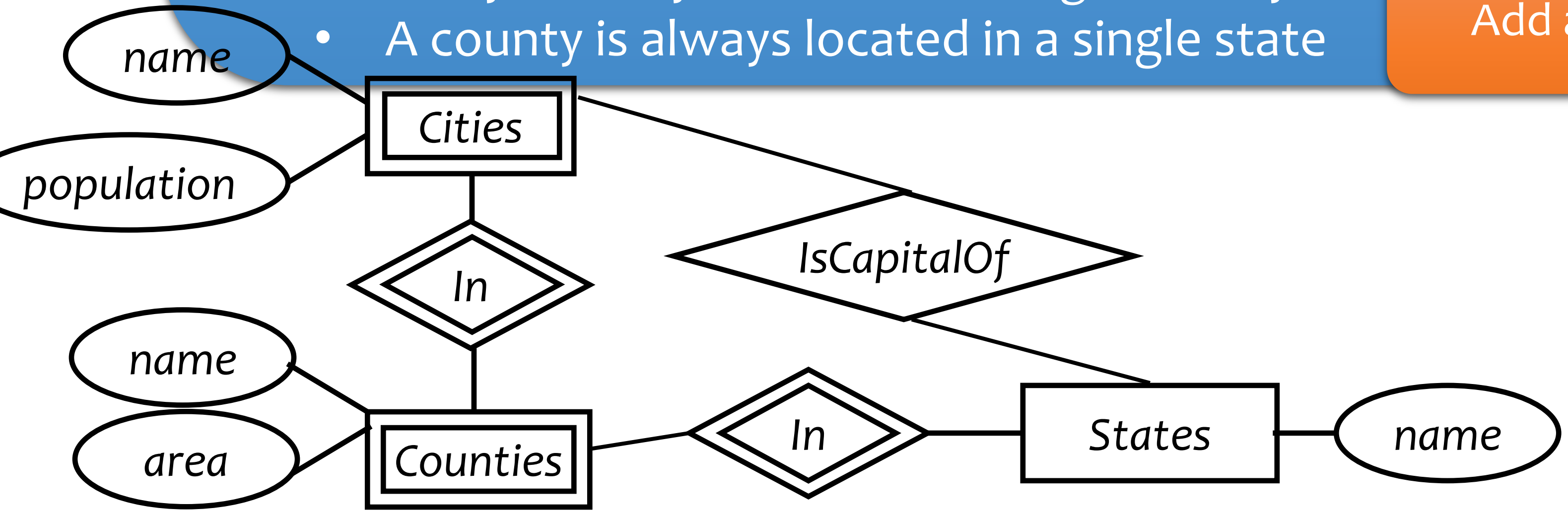
Design a database representing cities, counties, and states

- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

Assume the following:

- Names of states are unique
- Names of counties are only unique within a state
- Names of cities are only unique within a county
- A city is always located in a single county
- A county is always located in a single state

Add attributes!



Case study 1

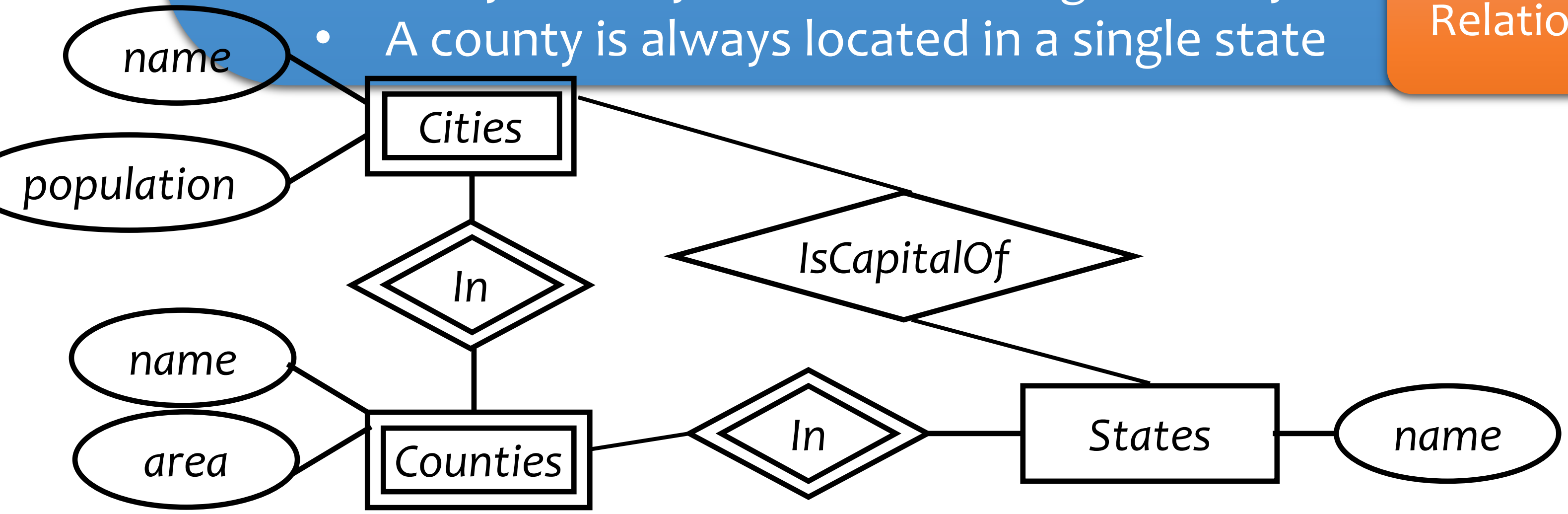
Design a database representing cities, counties, and states

- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

Assume the following:

- Names of states are unique
- Names of counties are only unique within a state
- Names of cities are only unique within a county
- A city is always located in a single county
- A county is always located in a single state

Relationship types?



Case study 1

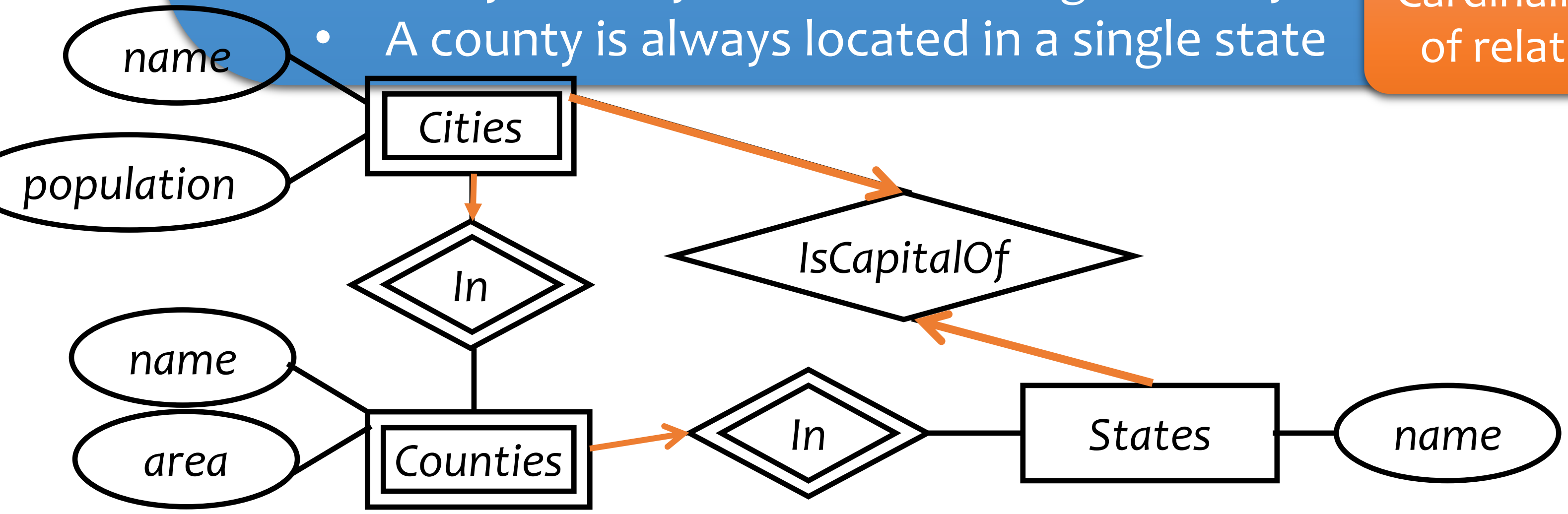
Design a database representing cities, counties, and states

- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

Assume the following:

- Names of states are unique
- Names of counties are only unique within a state
- Names of cities are only unique within a county
- A city is always located in a single county
- A county is always located in a single state

Cardinality constraints of relationship sets?



Case study 1

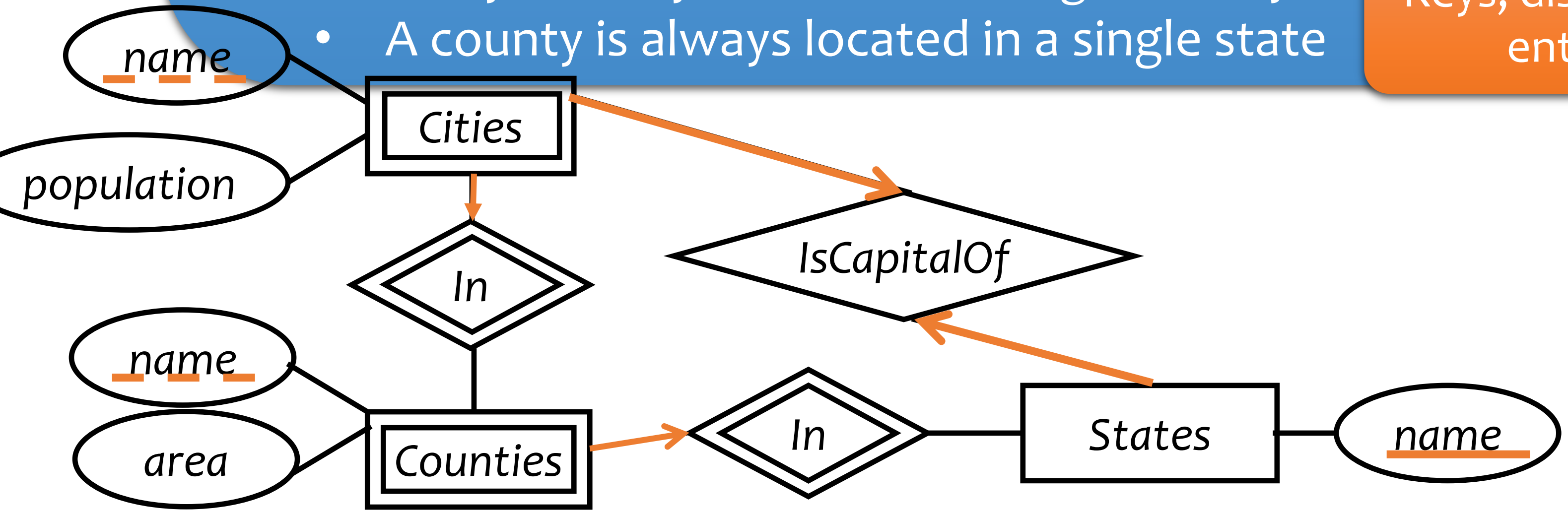
Design a database representing cities, counties, and states

- For states, record name and capital (city)
- For counties, record name, area, and location (state)
- For cities, record name, population, and location (county and state)

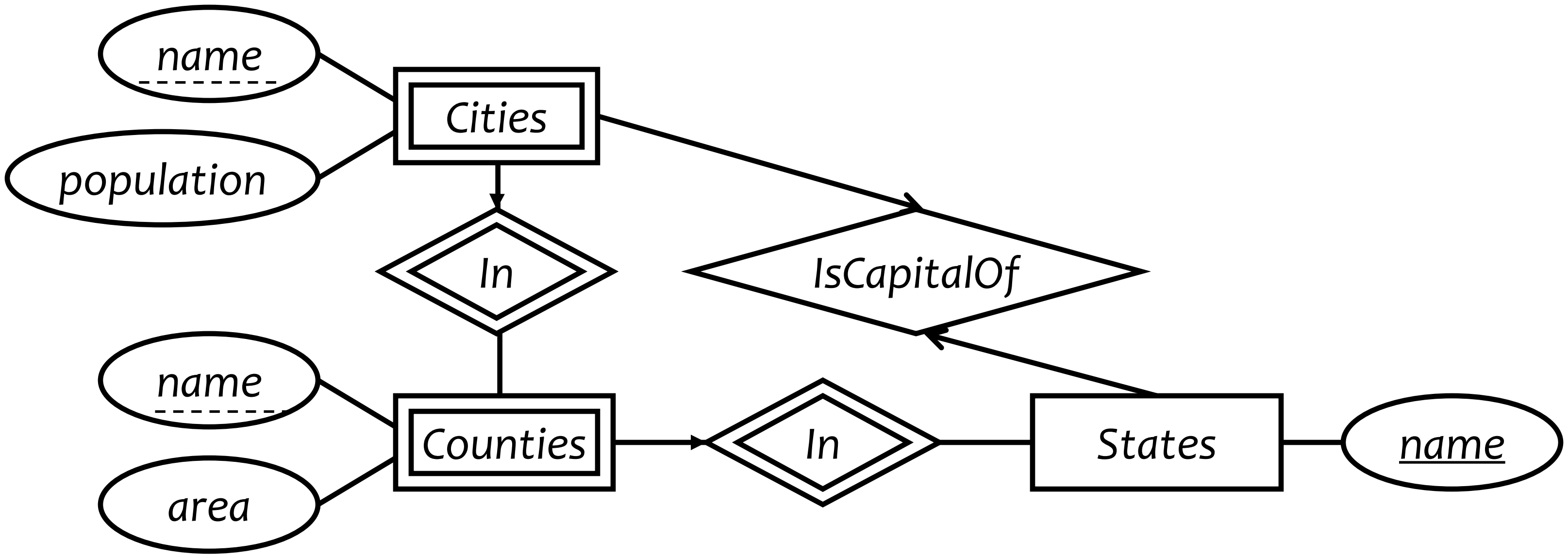
Assume the following:

- Names of states are unique
- Names of counties are only unique within a state
- Names of cities are only unique within a county
- A city is always located in a single county
- A county is always located in a single state

Keys, discriminator of entity sets?

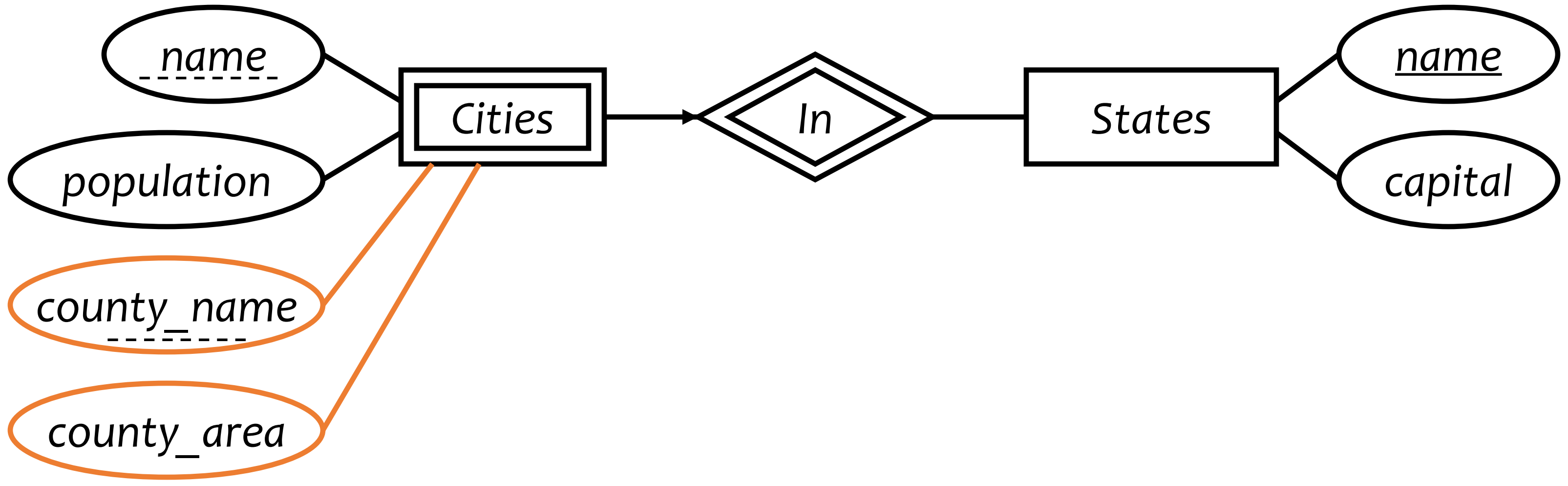


Case study 1: final design



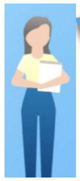
- Technically, nothing in this design prevents a city in state X from being the capital of another state Y, but oh well…

Case study 1: why not good?



- County area information is repeated for every city in the county
  - ☞Redundancy is bad (why?)
- State capital should really be a city
  - ☞Should "reference" entities through explicit relationships

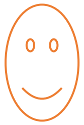
Case study 2



Design a database consistent with the following:

- A station has a unique name and an address, and is either an express station or a local station
- A train has a unique number and an engineer, and is either an express train or a local train
- A local train can stop at any station
- An express train only stops at express stations
- A train can stop at a station for any number of times during a day
- Train schedules are the same everyday

What are the entity sets, relationship sets, and their attributes? What are the types of relationships and cardinality constraints, keys, discriminators?



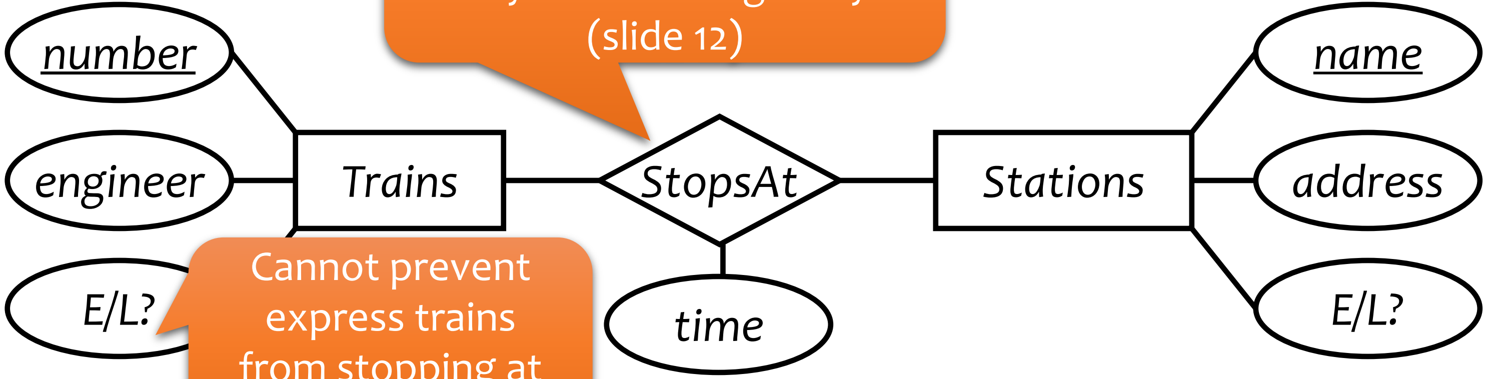
Case study 2: first design

Design a database consistent with the following:

- A station has a unique name and an address, and is either an express station or a local station
- A train has a unique number and an engineer, and is either an express train or a local train
- A local train can stop at any station
- An express train only stops at express stations
- A train can stop at a station for any number of times during a day
- Train schedules are the same everyday

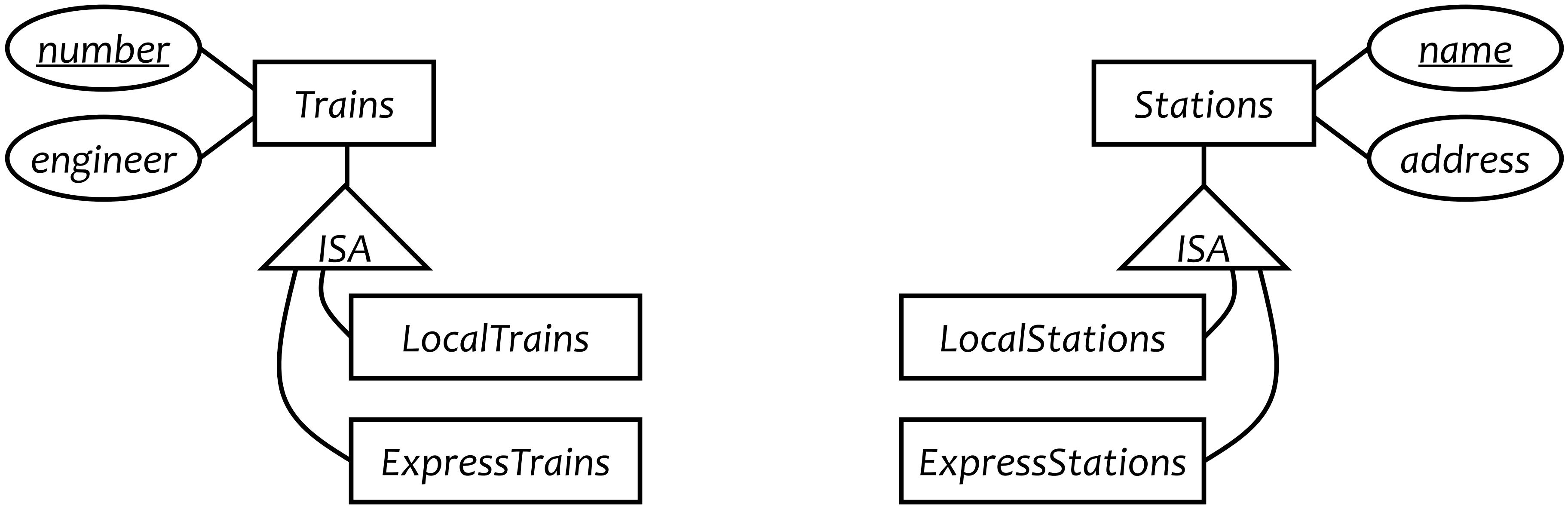
Why not good?

Unintended constraint: A train can stop at a station only ONCE during a day! (slide 12)



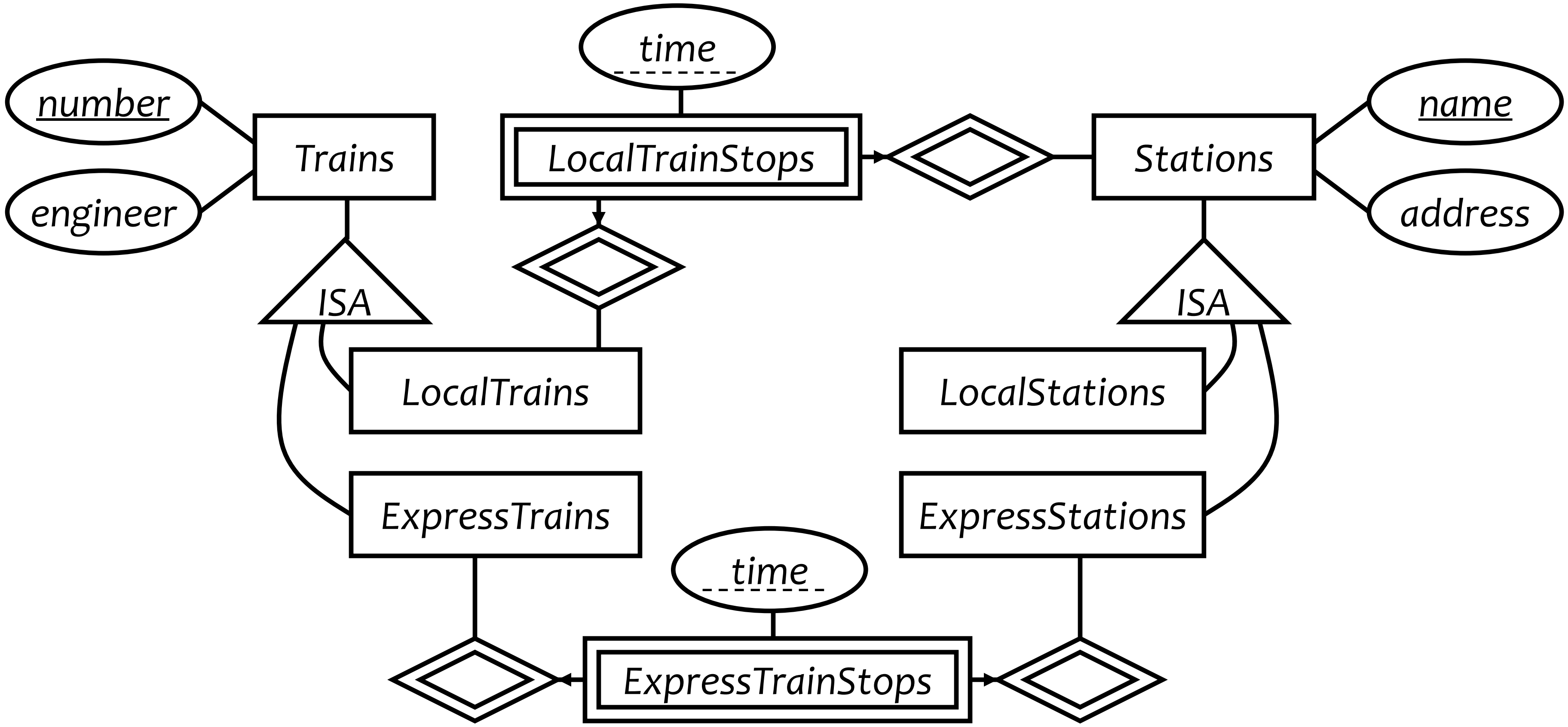
Cannot prevent express trains from stopping at any local stations

Case study 2: second design



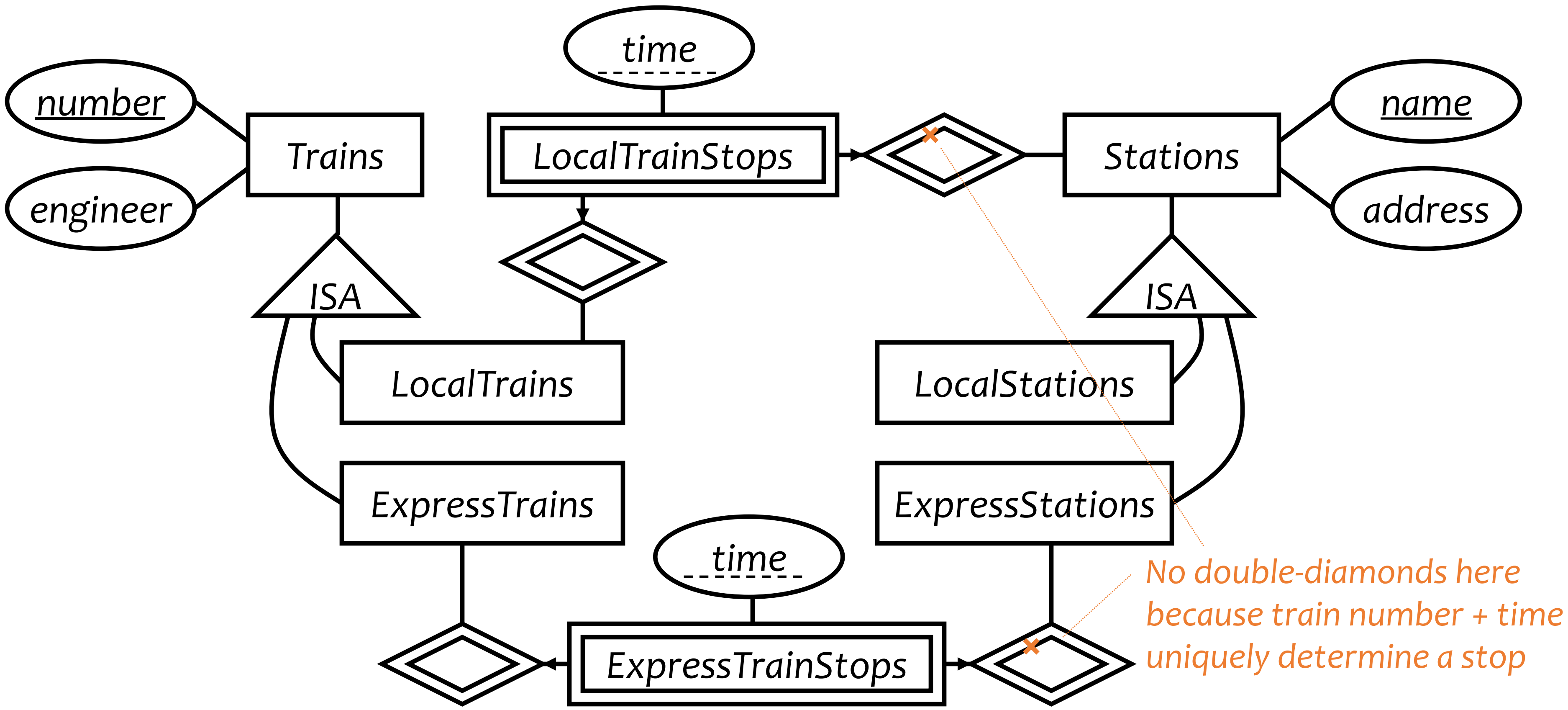
- A station has a unique name and an address, and is either an express station or a local station
- A train has a unique number and an engineer, and is either an express train or a local train
- … ..

Case study 2: second design



- …
- A local train can stop at any station
- An express train only stops at express stations
- A train can stop at a station for any number of times during a day
- …

Case study 2: second design



Is the extra complexity worth it?

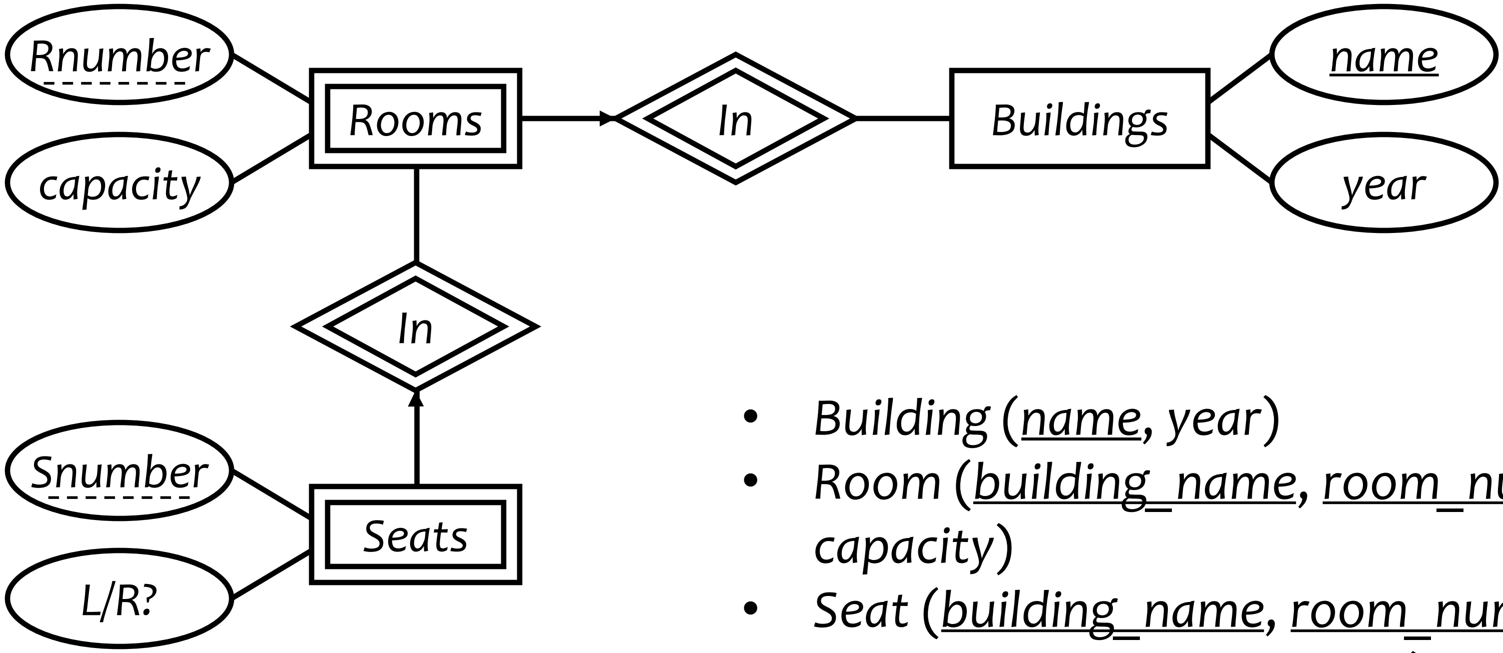
Case study 3 (Exercise)

- A Registrar's Database:
  - Zero or more sections of a course are offered each term. Courses have names and numbers. In each term, the sections of each course are numbered starting with 1.
  - Most course sections are taught on-site, but a few are taught at off-site locations.
  - Students have student numbers and names.
  - Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.
  - Up to 50 students may be registered for a course section. Sections with 5 or fewer students are cancelled.
  - A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.

What you have learned so far

- Entity-Relationship (E/R) model

- Next: Translating E/R to relational schema



- *Building (<u>name</u>, year)*
- *Room (<u>building_name</u>, <u>room_number</u>, capacity)*
- *Seat (<u>building_name</u>, <u>room_number</u>, <u>seat_number</u>, left_or_right)*

Solution for Case Study 3

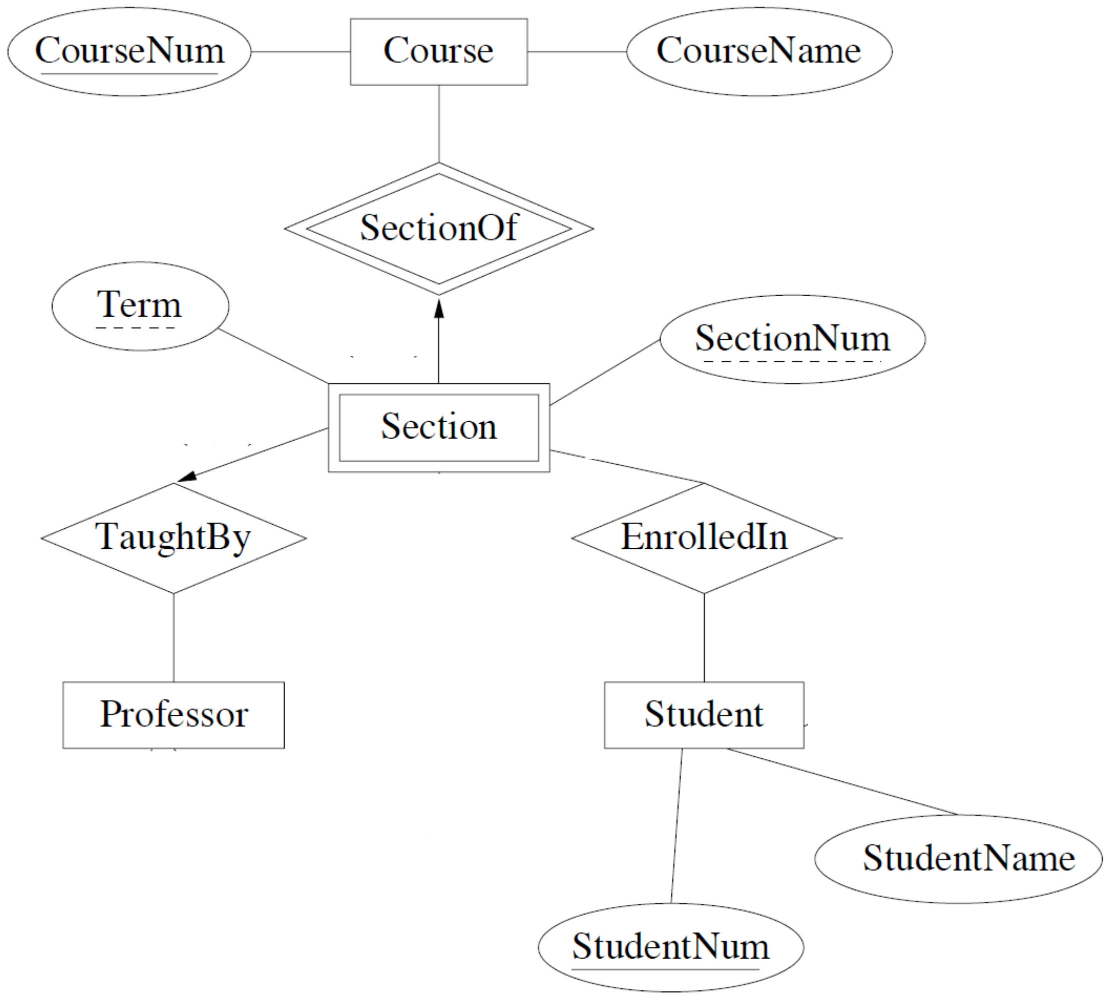
Case study 3 (Exercise)

- A Registrar's Database:
  - Zero or more sections of a course are offered each term. Courses have names and numbers. In each term, the sections of each course are numbered starting with 1.
  - Most course sections are taught on-site, but a few are taught at off-site locations
  - Students have student numbers and names.
  - Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.
  - Up to 50 students may be registered for a course section. Sections with 5 or fewer students are cancelled.
  - A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.

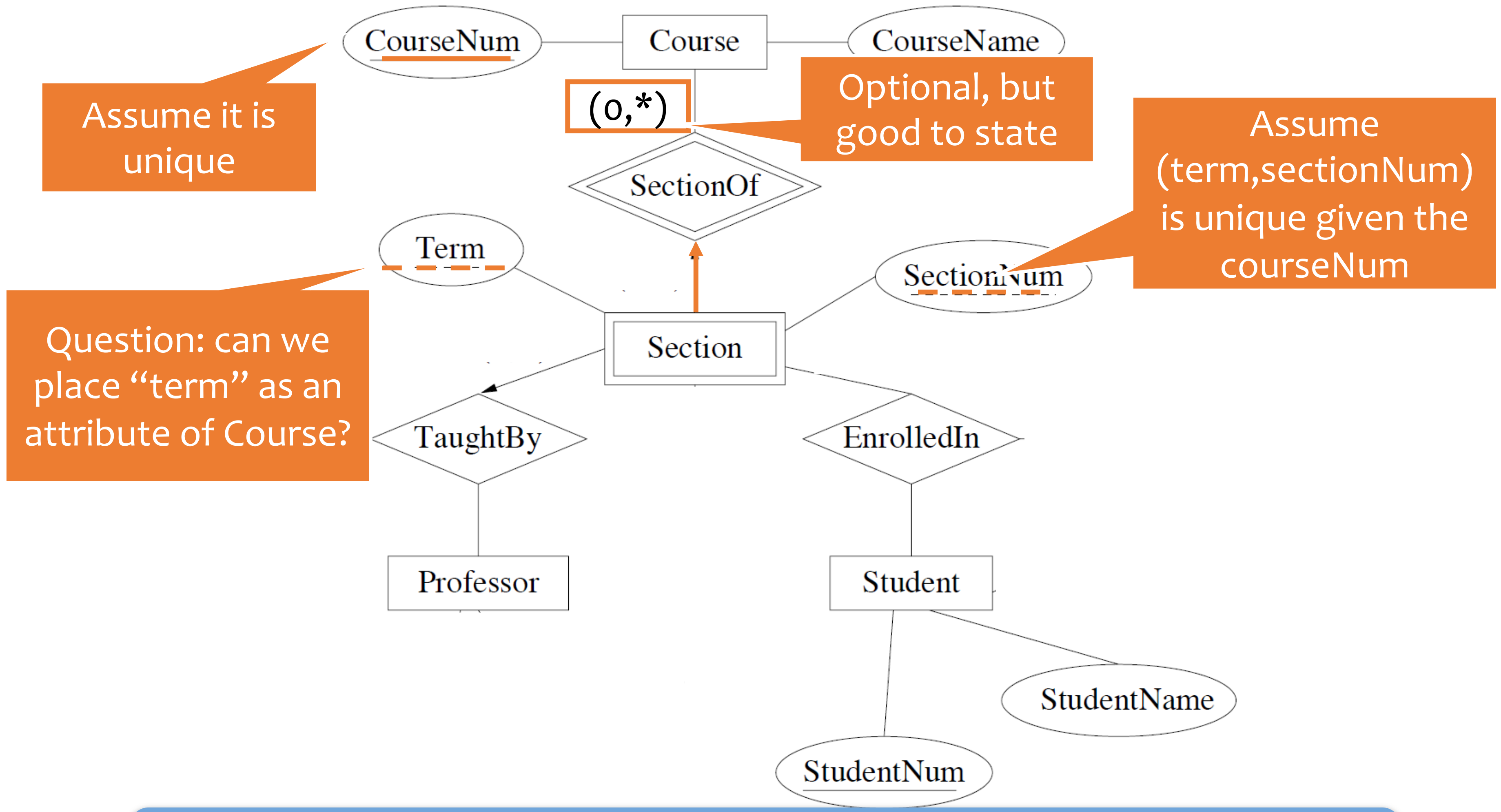
Case study 3 (Exercise)

- A Registrar's Database:
  - Zero or more sections of a course are offered each term. Courses have names and numbers. In each term, the sections of each course are numbered starting with 1.
  - Most course sections are taught on-site, but a few are taught at off-site locations.
  - Students have student numbers and names.
  - Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.
  - Up to 50 students may be registered for a course section. Sections with 5 or fewer students are cancelled.
  - A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.

Case study 3 (Exercise) cont.



Case study 3 (Exercise) cont.



Assume it is unique

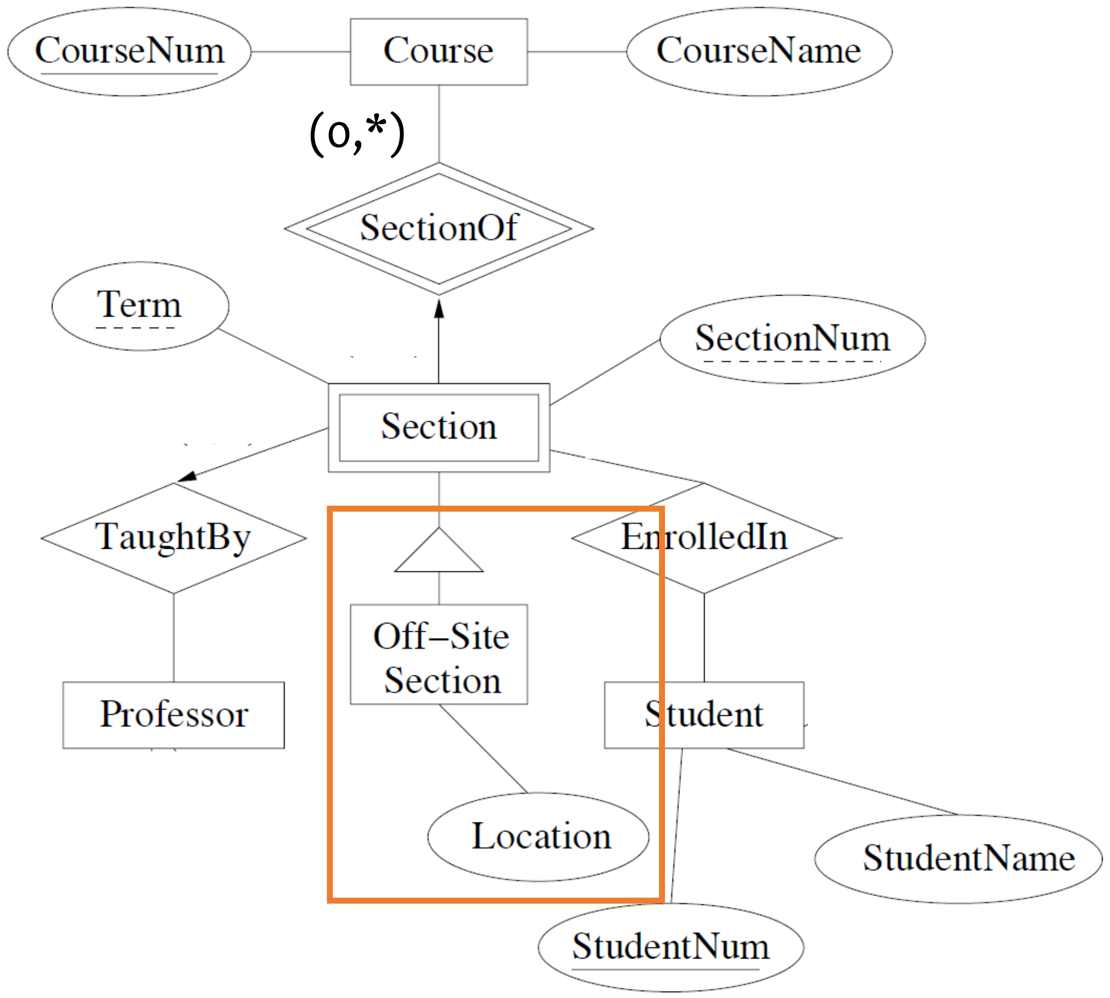
Optional, but good to state

Assume (term,sectionNum) is unique given the courseNum

Question: can we place "term" as an attribute of Course?

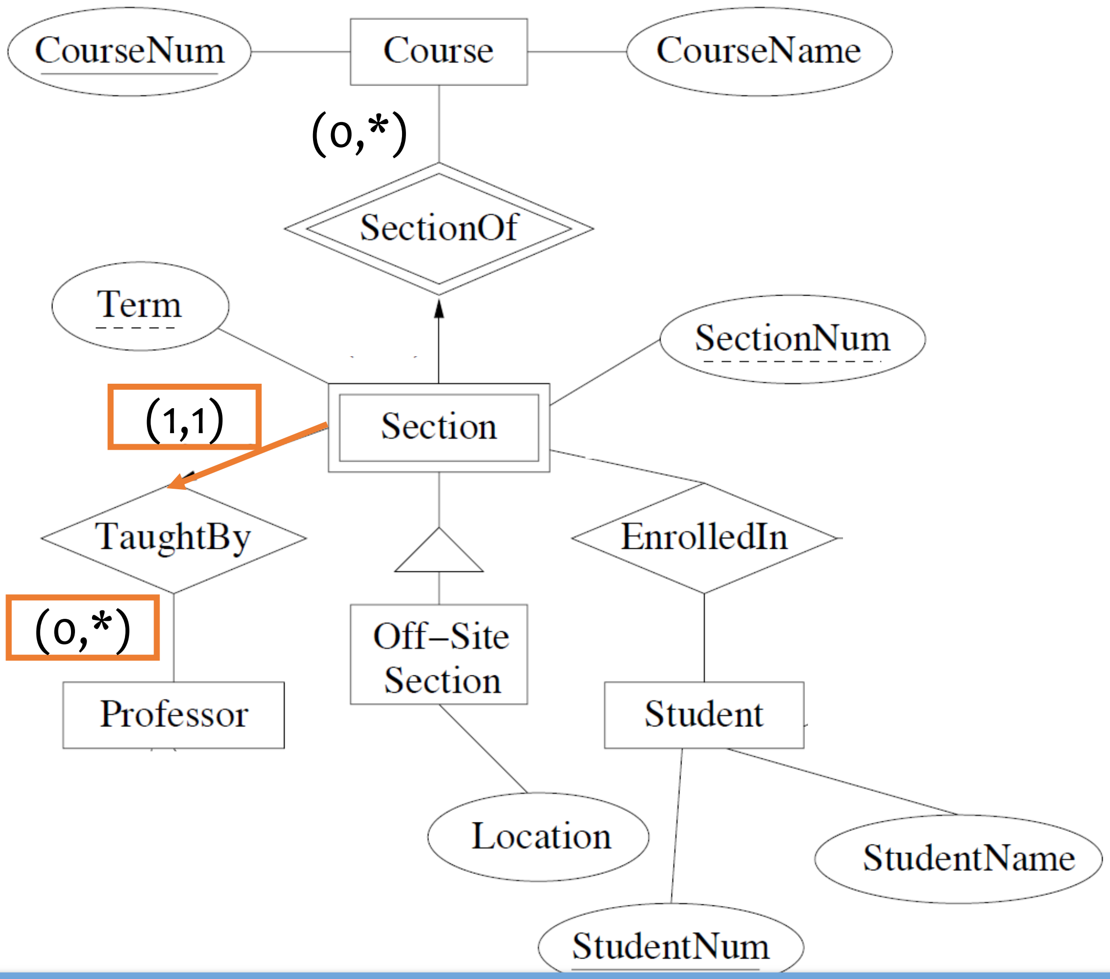
Zero or more sections of a course are offered each term. Courses have names and numbers. In each term, the sections of each course are numbered starting with 1.

Case study 3 (Exercise) cont.



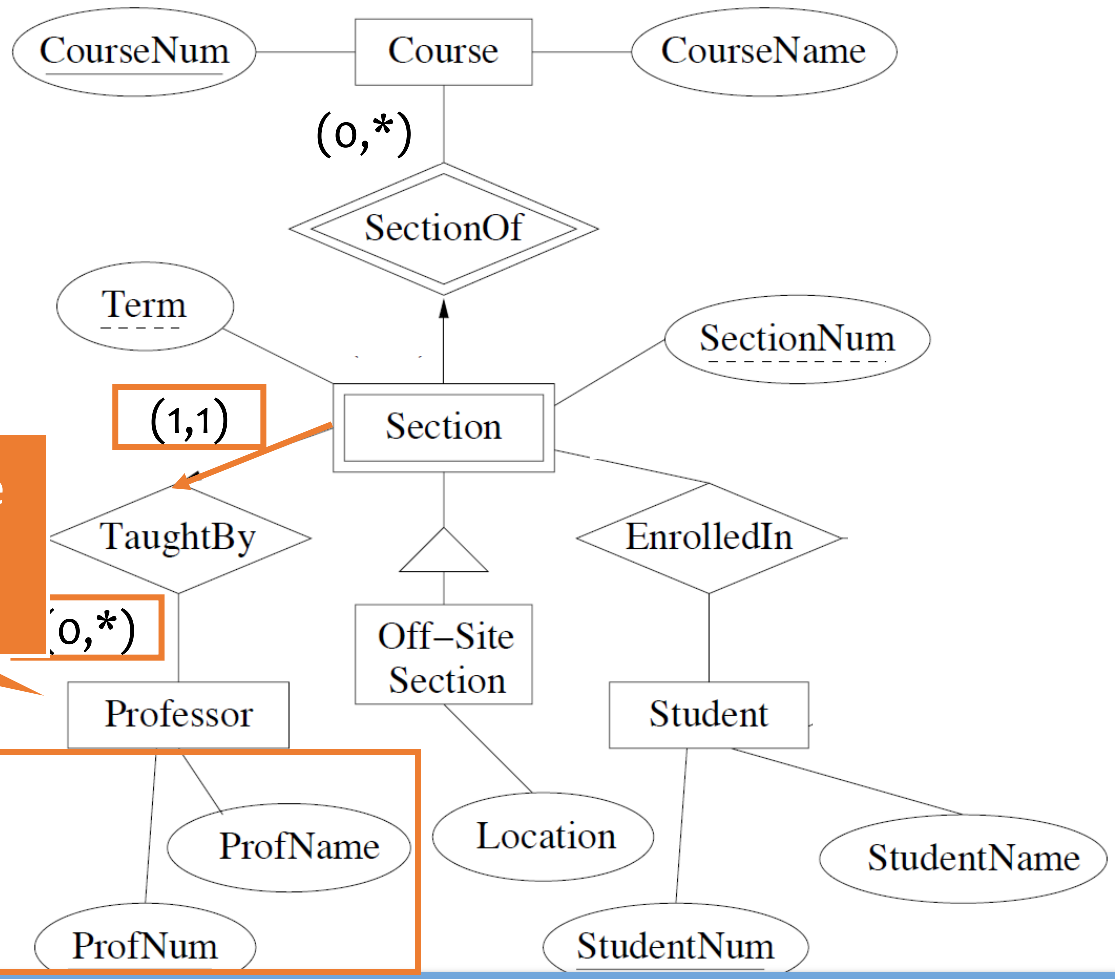
Most course sections are taught on-site, but a few are taught at off-site locations.

Case study 3 (Exercise) cont.



Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.

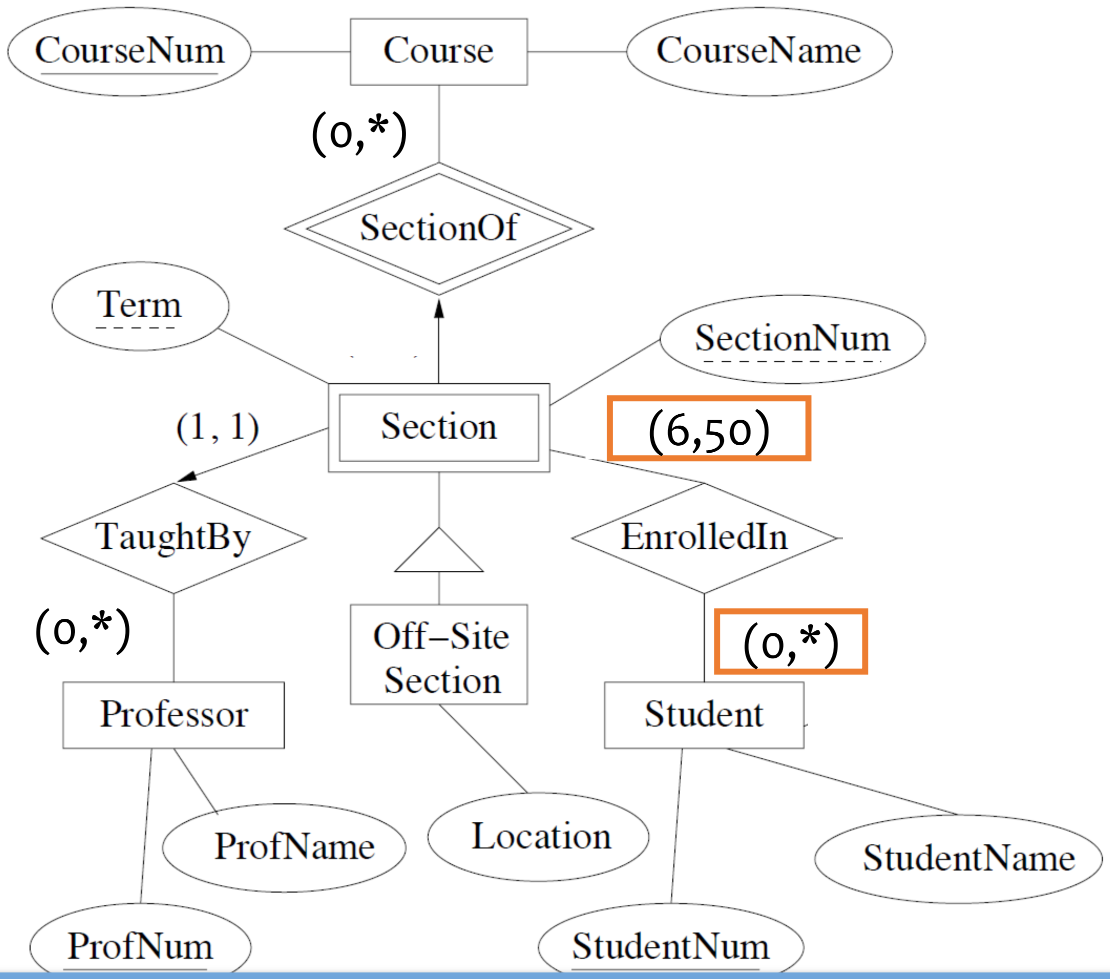
Case study 3 (Exercise) cont.



Can "Professor" be an attribute of Section?

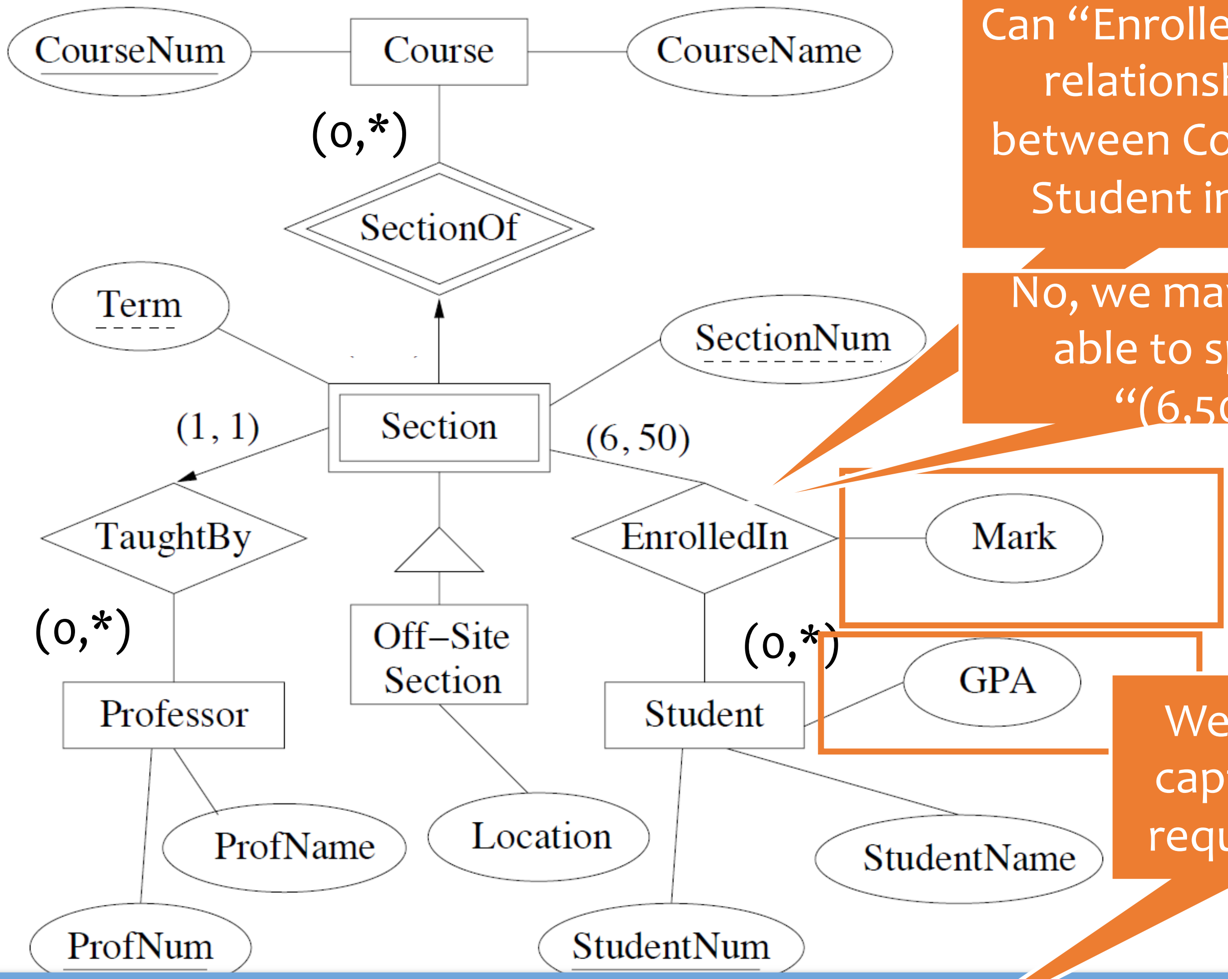
Each course section is taught by a professor. A professor may teach more than one section in a term, but if a professor teaches more than one section in a term, they are always sections of the same course. Some professors do not teach every term.

Case study 3 (Exercise) cont.



Up to 50 students may be registered for a course section. Sections with 5 or fewer students are cancelled.

Case study 3 (Exercise) cont.



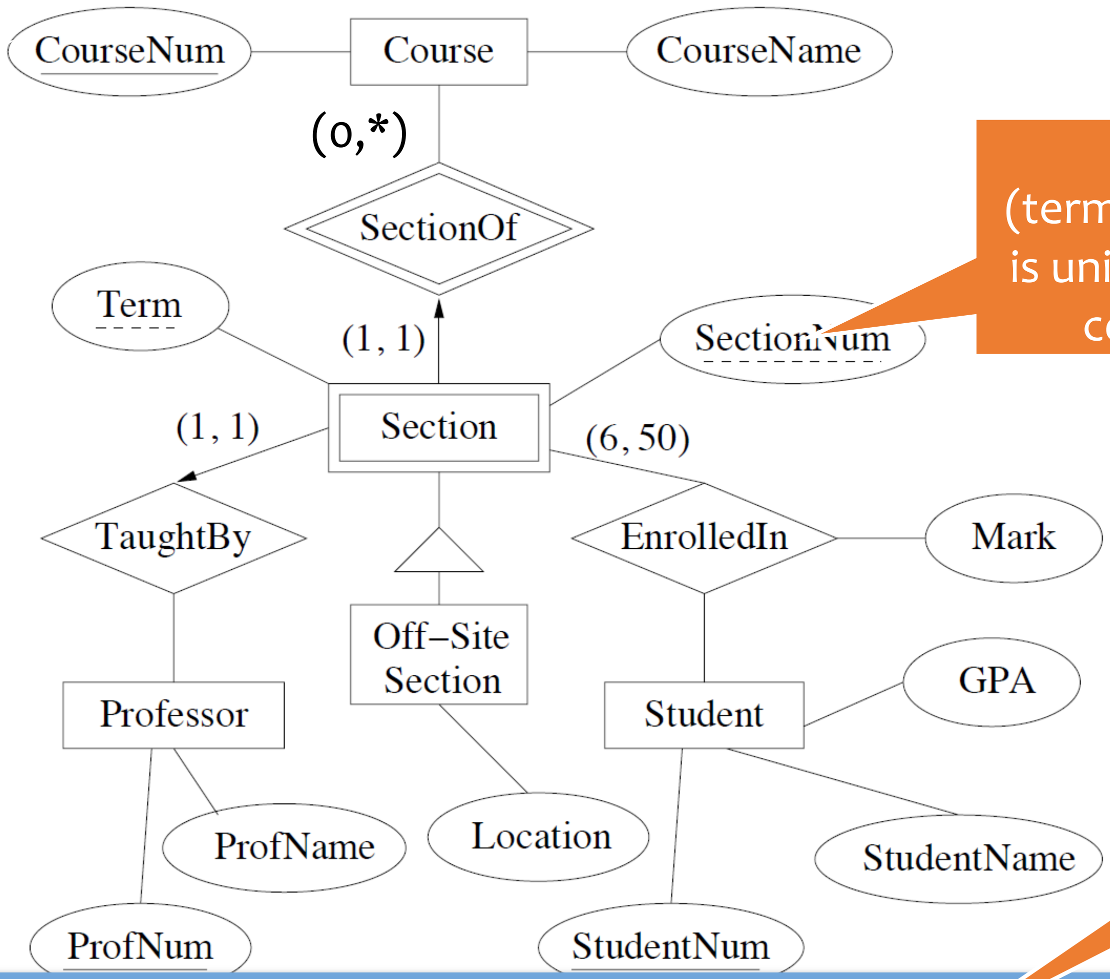
Can "EnrolledIn" be a relationship set between Course and Student instead?

No, we may not be able to specify "(6,50)"

We did not capture this requirement

A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.

Case study 3: possible solution



Assume it is unique

Assume (term,sectionNum) is unique given the courseNum

We did not capture this requirement

A student receives a mark for each course in which they are enrolled. Each student has a cumulative grade point average (GPA) which is calculated from all course marks the student has received.